Departamento de Señales, Sistemas y Radiocomunicaciones
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid

SWARM INTELLIGENCE: NOVEL TOOLS FOR OPTIMIZATION, FEATURE EXTRACTION, AND MULTI-AGENT SYSTEM MODELING

TESIS DOCTORAL

Aleksandar Jevtić
Ingeniero en Electrónica

2011

Departamento de Señales, Sistemas y Radiocomunicaciones
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid

SWARM INTELLIGENCE: NOVEL TOOLS FOR OPTIMIZATION, FEATURE EXTRACTION, AND MULTI-AGENT SYSTEM MODELING

TESIS DOCTORAL

Autor:

Aleksandar Jevtić

Ingeniero en Electrónica

Director:

Diego Andina de la Fuente

Doctor Ingeniero de Telecomunicación

Co-Director:

Mo Jamshidi

Ph.D. in Electrical Engineering

2011

Tribunal nombrado por el Mgfco. y Excmo. Sr. Rector de la Universidad Politécnica de Madrid, el día _____ de _____ de 2011.

Presidente: Dr. D. _____

Secretario: Dr. D. _____

Vocal: Dr. D. _____

Vocal: Dr. D. _____

Vocal: Dr. D. _____

Suplente: Dr. D. _____

Suplente: Dr. D. _____

Realizando el acto de defensa y lectura de la Tesis el día _____ de _____ de 2011.

En la E.T.S. de Ingenieros de Telecomunicación.

CALIFICACIÓN:

EL PRESIDENTE LOS VOCALES

EL SECRETARIO

Abstract

Animal swarms in nature are able to adapt to dynamic changes in their environment, and through cooperation they can solve problems that are crucial for their survival. Only by means of local interactions with other members of the swarm and with the environment, they can achieve a common goal more efficiently than it would be done by a single individual. This problem-solving behavior that results from the multiplicity of such interactions is referred to as Swarm Intelligence. The mathematical models of swarming behavior in nature were initially proposed to solve optimization problems. Nevertheless, this decentralized approach can be a valuable tool for a variety of applications, where emerging global patterns represent a solution to the task at hand. Methods for the solution of difficult computational problems based on Swarm Intelligence have been experimentally demonstrated and reported in the literature. However, a general framework that would facilitate their design does not exist yet.

In this dissertation, a new general design methodology for Swarm Intelligence tools is proposed. By defining a discrete space in which the members of the swarm can move, and by modifying the rules of local interactions and setting the adequate objective function for solutions evaluation, the proposed methodology is tested in various domains. The dissertation presents a set of case studies, and focuses on two general approaches. One approach is to apply Swarm Intelligence as a tool for optimization and feature extraction, and the other approach is to model multi-agent systems such that they resemble swarms of animals in nature providing them with the ability to autonomously perform a task at hand.

Artificial swarms are designed to be autonomous, scalable, robust, and adaptive to the changes in their environment. In this work, the methods that exploit one or more of these features are presented. First, the proposed methodology is validated in a real-world scenario seen as a combinatorial optimization problem. Then a set of novel tools for feature extraction, more precisely the adaptive edge detection and the broken-edge linking in digital images is proposed. A novel data clustering algorithm is also proposed and applied to image segmentation. Finally, a scalable algorithm based on the proposed methodology is developed for distributed task allocation in multi-agent systems, and applied to a swarm of robots. The newly proposed general methodology provides a guideline for future developers of the Swarm Intelligence tools.

Resumen

Los enjambres de animales en la naturaleza son capaces de adaptarse a cambios dinámicos en su entorno y, por medio de la cooperación, pueden resolver problemas cruciales para su supervivencia. Únicamente por medio de interacciones locales con otros miembros del enjambre y con el entorno, pueden lograr un objetivo común de forma más eficiente que lo haría un solo individuo. Este comportamiento problema-resolutivo que es resultado de la multiplicidad de interacciones se denomina Inteligencia de Enjambre. Los modelos matemáticos de comportamiento de enjambres en entornos naturales fueron propuestos inicialmente para resolver problemas de optimización. Sin embargo, esta aproximación descentralizada puede ser una herramienta valiosa en una variedad de aplicaciones donde patrones globales emergentes representan una solucin de las tareas actuales.

Aunque en la literatura se muestra la utilidad de los métodos de Inteligencia de Enjambre, no existe un entorno de trabajo que facilite su diseño. En esta memoria de tesis proponemos una nueva metodologa general de diseño para herramientas de Inteligencia de Enjambre. Desarrollamos herramientas noveles que representan ejemplos ilustrativos de su implementación. Probamos la metodología propuesta en varios dominios definiendo un espacio discreto en el que los miembros del enjambre pueden moverse, modificando las reglas de las interacciones locales y fijando la función objetivo adecuada para evaluar las soluciones. La memoria de tesis presenta un conjunto de casos de estudio y se centra en dos aproximaciones generales. Una aproximación es aplicar Inteligencia de Enjambre como herramienta de optimización y extracción de características mientras que la otra es modelar sistemas multi-agente de tal manera que se asemejen a enjambres de animales en la naturaleza a los que se les confiere la habilidad de ejecutar autónomamente la tarea.

Los enjambres artificiales están diseñados para ser autónomos, escalables, robustos y adaptables a los cambios en su entorno. En este trabajo, presentamos métodos que explotan una o más de estas características. Primero, validamos la metodología propuesta en un escenario del mundo real visto como un problema de optimización combinatoria. Después, proponemos un conjunto de herramientas noveles para extracción de características, en concreto la detección adaptativa de bordes y el enlazado de bordes rotos en imágenes digitales, y el agrupamiento de datos para segmentación de imágenes. Finalmente, proponemos un algoritmo escalable para la asignación distribuida de tareas en sistemas multi-robots. La metodología general propuesta ofrece una guía para futuros desarrolladores de herramientas de Inteligencia de Enjambre.

To my family

Acknowledgments

During the years of my Ph.D. studies, many people have formed an important part of my life and to all of them I wish to express my sincerest gratitude for the support they have given me.

I thank my advisor Professor Diego Andina de la Fuente for the opportunity to be a part of the Group for Automation in Signal and Communications. I am grateful for his guidance and his continuing support, and for allowing me to choose the research topic of my interest. In his words: "It is important that you choose a topic that interests you, because you are the one that will have to dedicate years to work on it".

I also thank my co-advisor Professor Mo Jamshidi for his support and professional advice in the final years of my studies. During my visit to the University of Texas at San Antonio, I had the opportunity to collaborate with the researchers from his Autonomous Control Engineering center, to all of whom I would like to express my thanks for accepting me as an equal member of the "ACE swarm".

The collaboration with the colleagues from my research group has greatly contributed to this work, and unlimited thanks go to Fulgencio Buendia, Sofia Corvera, Alexis Marcano, Miguel Barrón, Benjamín Ojeda, Joel Quintanilla, Guadalupe Cortina Januchs, Juan Fombellida and Ignacio Melgar. I would also like to express my gratitude to professors Joaquín Torres, Ana Tarquis, Juan Grau, Juan Seijas and Antonio Vega, and Mr. Carlos González Mateos for their support.

Numerous thanks to Dr. Álvaro Gutiérrez for his unconditional help with my research work and for the fruitful research-related discussions that we had. Special thanks to my friends and colleagues Nelson Dopico and Carlos R. del Blanco, for sharing the Ph.D. challenges with me from the very beginning, and to all the people from their research groups that I had a pleasure of meeting.

My sincerest gratitude goes to the professors from the Universidad Politécnica de Madrid that I had a pleasure of learning from. I would also like to thank to the administration staff of the E.T.S.I. de Telecomunicación, Maite, Mari Carmen and Cristina, and of the Universidad Politécnica de Madrid, Silvia, Carolina, and Carmen, for making the administrative part of my work as easy as it can get.

During my visits to other universities, I had the opportunity of working with many great researchers. I would like to thank Professor Duc Truong Pham for inviting me to visit on several occasions the Manufacturing Engineering Centre at the Cardiff University. It is in the laboratories of this center that I got the first insight into Swarm Intelligence, which triggered my interest in this field. I would also like to

thank Dr. Michael Packianather, Dr. Nikolay Zlatov, Dr. Ashraf Fahmy, Dr. Ming Yang, Dr. Ashraf Afify, and many others who I had a pleasure of collaborating with during my stays in Cardiff.

Many thanks to Dr. Marco Castellani, with whom I share many research interests and who had enough patience to answer my questions and with the same patience to listen to my ideas and expose them to critical and creative thinking. Unlimited thanks to Moises Sanchez for being a great friend and an admirable host during my visits to Cardiff.

I would like to thank Professor Željko Djurović for inviting me to work as a visiting lecturer at the University of Belgrade, and for his advice and support during the last year of my doctoral studies.

Many thanks to Professor Filip Kulić for giving me the opportunity to visit and work at the Faculty of Technical Sciences in Novi Sad. I would especially like to thank Mr. Miloš Jovanović who took care of all the administrative details of my visit, and for being a great friend and an excellent host.

Deepest thanks to Professor Darko Marinov, my long-time friend and someone I constantly learn from, for his support from the beginning of my undergraduate studies. Many thanks to Professor Slobodan Bojanić for great discussions and for being a patient tennis instructor.

I would especially like to thank Miloš and Marina Jakovljević for the great friendship that we have had for so many years, and for supporting me to come to Madrid. Unlimited thanks to my dearest friends Vučko, Steva, Jelena Jovanović, Miguel, Cristina Montero, Jose, Anne, Edu, Paloma, Filip, Cristina Cabrero, Marko, Andrijana, Goran, Dejana, Milena, Branko, Srdjan, Sara, Ruben, Ale, Nazario, Mladen, Tamara, Višnja, Milan, Paola, Jelena Novaković, Rade, Jelena Lazarević, Mateja, Luka, Lia, Bordallo, and others that shared this journey with me.

Finally, the ones whose unconditional love and support marks everything I do are my parents Rade and Ljiljana, and my sister Nataša and her husband Saša. They have the best and the worst of me, and they have been the light of my life. This thesis is dedicated to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The world around us is becoming increasingly complex every day and changes dynamically. The problems that we face require adaptive and scalable systems that can offer solutions with ever-rising level of autonomy. Traditional approaches are becoming obsolete because they were designed for a simpler world. Therefore, any advancement in understanding and solving of complex problems can have an impact on the entire set of disciplines in engineering, biology, sociology, etc.

Swarm Intelligence is a problem-solving behavior that occurs as a result of a multiplicity of interactions between independent components that make up the entire system, i.e. the swarm. The algorithms inspired by the cooperative behavior in nature, as in colonies of social insects, rely on artificial swarms of agents and were initially applied to solve the combinatorial optimization problems. These algorithms are iterative computational methods and offer better solutions at the expense of longer computation time. The individual agents in the swarm are not aware of the global objective, thus by modifying the rules of local interactions the algorithm can be customized to solve different problems.

Many mathematical models of Swarm Intelligence have been proposed in literature, and in most cases they offer a solution to a specific problem or a group of similar problems without an attempt to create a general framework for their design. In this dissertation, various methods are proposed that are based on the same rules of interaction between the members of the swarm, within a specific discrete environment, where the resulting swarming behavior is used as a tool for optimization and feature extraction, or as a model for multi-agent systems that resemble the real swarms in

nature. It is demonstrated how the multiplicity of interactions on the local level produces the global patterns that in different environments represent solutions to the above-mentioned problems.

1.2 Preview of Contributions

The algorithms proposed in this dissertation exploit the self-organizing behavior of a large group of autonomous agents to efficiently solve the above-mentioned problems by providing scalability, robustness, and/or adaptability. The main objective was to propose a general design methodology for Swarm Intelligence-based methods. The focus was made on two main approaches. One approach was to apply Swarm Intelligence as a tool, i.e. the underlying mechanism of a system, which based on the input values produces the output that represents a solution to a given problem. The second approach was to apply Swarm Intelligence to model the behavior of the system made up of a large number of autonomous subsystems (agents) in order to achieve some expected problem-solving behavior.

This dissertation presents the research work on the novel methods of Swarm Intelligence for various application domains. It demonstrates the advantages of a decentralized, bottom-up approach in optimization, feature extraction, and multi-agent system modeling. The original contributions are made in the areas of design methodology and application of the Swarm Intelligence methods. More specifically, the contributions of this dissertation are as follows:

- **Definition of the general methodology for the design of the Swarm Intelligence tools (Chapter 3)**: The general design methodology consists in defining the discrete data space in which the members of the swarm move, the rules of local interactions, and the objective function for solutions evaluation. It provides a unified probabilistic rule for transition between the neighboring nodes in the data space.

- **Validation of the proposed swarm-based methodology (Chapter 4)**: The proposed swarm-based methodology was validated in the real-world scenario, as a tool for the path optimization of Unmanned Aerial Vehicles (UAVs). The solution produced by the Ant System algorithm was combined with the dynamical model of the UAV in order to find the most cost-effective path for UAVs in the scenario of area coverage with a predefined set of waypoints.

  Published in:

Jevtić, A., Andina, D., Jaimes, A., Gomez, J., and Jamshidi, M. (2010). Unmanned aerial vehicle route optimization using Ant System algorithm. In *Proceedings of the 5th IEEE International Conference on System of Systems Engineering, SoSE 2010*, pages 1-6.

- **Novel edge-detection method (Chapter 5)**: An edge detection method was proposed that combines a nonlinear contrast enhancement technique, called Multiscale Adaptive Gain, and the Ant System algorithm, which is based on the ant foraging behavior. The set of enhanced images obtained after applying the Multiscale Adaptive Gain and the Ant System algorithm is used to create pheromone patterns where the true edges are found. The sum of the obtained pheromone matrices, after applying threshold and morphological thinning, produces the output edge image. The proposed method was more efficient in finding well-connected edges, in which it outperformed other state-of-the-art ant colony-based edge detectors.

  Published in:

  Jevtić, A., Quintanilla-Domínguez, J., Cortina-Januchs, M. G., and Andina, D. (2009). Edge detection using ant colony search algorithm and multiscale contrast enhancement. In *Proceedings of the 2009 IEEE International Conference on Systems, Man, & Cybernetics, SMC 2009*, pages 2193-2198.

- **Adaptive edge detection (Chapter 5)**: The adaptability of the above-mentioned edge detector was demonstrated in a dynamically changing environment made of a set of digital grayscale images. The algorithm responds to the changes by creating different pheromone patterns according to the distribution of the newly-created edges. It also shows to be robust, because any smaller artificial ant colony manages to detect the edges even though the total number of the detected edges is reduced.

  Published in:

  Jevtić, A. and Andina, D. (2010). Adaptive artificial ant colonies for edge detection in digital images. In *Proceedings of the 36th Annual Conference on IEEE Industrial Electronics Society, IECON 2010*, pages 2813-2816.

- **Novel broken-edge linking method (Chapter 5)**: Broken edge linking is an image improvement technique that is complementary to edge detection, where the broken edges are connected to form closed contours in order to separate the regions in the image. A method based on the Ant System algorithm

was developed in which the artificial ants search for the edge segments that connect different endpoints. A novel fitness function for solution evaluation was introduced, which is dependent on two variables: the grayscale visibility of the pixels, and the length of the connecting edge segment. The fitness function made more favorable the segments that consisted of smaller number of pixels, which had grayscale visibility of a higher mean value and a lower variance. Another introduced novelty was to apply the grayscale visibility matrix as the initial pheromone trails matrix so that the pixels belonging to true edges have a higher probability of being chosen by ants on their initial routes, which reduced computational load. The proposed broken-edge linking method was tested as a complementary tool for the Sobel edge detector, and it significantly improved the output edge image.

Published in:

Jevtić, A., Melgar, I., and Andina, D. (2009). Ant based edge linking algorithm. In *Proceedings of the 35th Annual Conference of the IEEE Industrial Electronics Society, IECON 2009*, pages 3353-3358.

- **Novel data-clustering algorithm (Chapter 6)**: The Ant System-based Clustering Algorithm (ASCA), a data-clustering algorithm that models the pheromone-laying, pheromone-following behavior of natural ant colonies was proposed in order to extract the number of clusters and subsequently classify data patterns by assigning them to these clusters. It showed high sensitivity in detection of small clusters, i.e. the atypical data, that are in the proximity of larger clusters. The algorithm was applied to image segmentation, for detection of microcalcifications in digital mammograms, and it outperformed other state-of-the-art algorithms that it was compared to, such as 1D-SOM, k-Means, Fuzzy c-Means and Possibilistic Fuzzy c-Means. The important feature of the ASCA algorithm to extract the number of clusters is extremely useful for applications where the groups of patterns within a data set are not well-defined and need to be detected.

  Published in:

  Jevtić, A., Quintanilla-Domínguez, J., Barrón-Adame, J. M., and Andina, D. (2011). Image segmentation using Ant System-based Clustering Algorithm. In *International Conference on Soft Computing Models in Industrial & Environmental Applications, SOCO 2011*. Accepted for publication.

- **Novel algorithm for distributed task allocation (Chapter 7)**: The Distributed Bees Algorithm (DBA) was proposed for a distributed task allocation in a swarm of autonomous mobile robots engaged in multi-foraging scenario. The algorithm was inspired by the foraging behavior of bees in nature, and it applies a bottom-up approach that provides the multi-robot system with autonomous decision-making. The algorithm achieves the objective of assigning the robots to the found targets in a way that the final distribution is proportional to the targets' quality (fitness) values. The algorithm was validated through experiments with real robots in the laboratory environment. The performance of the robot swarm was tested with respect to the odometry error and the random target search time. The scenario success criterium was established, and the time threshold value for its achievement was obtained.

  Published in:

  Jevtić, A., Gazi, P., Andina, D., and Jamshidi, M. (2010b). Building a swarm of robotic bees. In 2010 World Automation Congress, WAC 2010, pages 1-6. **Best Paper Award.**

- **Scalable multi-agent system (Chapter 7)**: The scalability of the abovementioned DBA algorithm was tested in a simulator, with respect to the number of robots and the number of tasks (targets) at hand. It was shown that the proposed multi-robot system is scalable, as its performance improved when more robots were engaged in the target search. The experiments were repeated for different sets of targets, having different total number of targets and by changing the targets' quality values. Although the experiments involve a multi-robot system, the DBA algorithm can be applied to a general problem of distributed task allocation by defining the qualities and the costs of all the tasks.

  In preparation to be published in:

  Jevtić, A., Gutiérrez-Martín, A., Andina, D., and Jamshidi, M. (2011). Distributed Bees Algorithm for task allocation in swarm of robots. *IEEE Systems Journal*. In preparation.

During the course of this work, significant contributions have also been made within the following projects:

- The Ant Colony Optimization metaheuristic for edge detection in digital images.

Jevtić, A., and Andina, D. (2011). Ant-based algorithms for digital image processing. Chapter in the book *Search Algorithms*, InTech Publisher. ISBN: 978-953-307-483-2. Accepted for publication.

- A framework for a distributed coordination of a swarm of robots using networked control algorithms.

  Gazi, P., Jevtić, A., Andina, D., and Jamshidi, M. (2010). A mechatronic system design case study: Control of a robotic swarm using networked control algorithms. In *Proceedings of the 4th Annual IEEE Systems Conference*, pages 169–173. **Best Paper Award.**

- Artificial Metaplasticity as a novel learning method for artificial neural networks.

  Andina, D, Alvarez-Vellisco, A., Jevtić, A., and Fombellida, J. (2009). Artificial metaplasticity can improve neural network learning. *Intelligent Automation and Soft Computing, Special Issue in Signal Processing and Soft Computing*, TSI Press, USA, 15(4):681–694. **JCR impact factor: 0.349.**

- Image segmentation using artificial neural networks and nonlinear filters.

  Quintanilla-Domínguez, J., Cortina-Januchs, M. G., Jevtić, A., Barrón-Adame, J. M., Vega-Corona, A., and Andina, D. (2009). Combination of nonlinear filters and ANN for detection of microcalcifications in digitized mammography. In *Proceedings of the IEEE International Conference on Systems, Man, & Cybernetics, SMC 2009*, pages 1516–1520.

- Review of the Swarm Intelligence algorithms and applications.

  Jevtić, A., and Andina, D. (2007). Swarm intelligence and its applications in swarm robotics. In *Proceedings of the 6th WSEAS International Conference on Computational Intelligence, Man-Machine Systems & Cybernetics, CIMMACS'07*, pages 41–46.

The study of the state-of-the-art Soft Computing methods applied to various domains was performed, which resulted in contribution in the following published work (Marcano-Cedeño et al., 2009; Andina et al., 2007; Quintanilla-Domínguez et al., 2010; Grau et al., 2009; Melgar et al., 2009; Alarcón-Mondéjar et al., 2008; Cortina-Januchs et al., 2008; Andina and Jevtić, 2007a,b).

1.3 Dissertation Outline

The rest of the dissertation is organized in four parts.

Part I presents an introduction to the field of Swarm Intelligence. Chapter 2 describes the biological inspiration from which the field of Swarm Intelligence originated. Chapter 3 gives an overview of the state-of-the-art algorithmic models of Swarm Intelligence, referred to as Computational Swarm Intelligence. In this chapter, a general design methodology for Swarm Intelligence tools is defined, and the probabilistic node transition rule is described as the underlying decision-making mechanism.

Part II describes the proposed Swarm Intelligence methods that are applied as tools for optimization and feature extraction. Three case studies are presented to illustrate the methodology and an overview of the related work for each application domain is given. Chapter 4 validates the general rule of local interaction on the problem of Unmanned Aerial Vehicle path optimization. Chapter 5 describes novel methods for edge detection and broken edge linking in digital grayscale images and performs the analysis of the adaptability of the proposed edge detector. Chapter 6 introduces a novel data clustering algorithm that is applied to image segmentation for the extraction of regions of the atypical pixels.

Part III discusses the application of Swarm Intelligence as a model for multi-agent systems and gives an overview of the related work in the field. Chapter 7 presents a case study of a distributed task allocation in a swarm of large number of autonomous mobile robots. The novel method exploits the foraging behavior of bees to model the recruitment of new robots for the most favorable targets. The proposed method was validated through experiments on the real robots. The analysis of the scalability of the proposed method is performed through experiments in a simulated environment by changing the size of the robot swarm and the number of targets.

Part IV summarizes the contributions of the thesis. Chapter 8 gives conclusions with an overview of the thesis' contributions and describes the possible lines of future work.

Part I

Background

Chapter 2

Biological Inspiration

2.1 Introduction

Swarm-based systems are typically made of a population of simple agents interacting locally with one another and with their environment to achieve a common goal. The benefits of cooperation can be significant in situations where global knowledge of the environment does not exist. Agents within the group interact by exchanging locally available information such that the global objective is obtained more efficiently than it would be done by a single agent. The group of agents acting in such a manner can be referred to as a swarm. The problem-solving behavior that emerges from the interactions of such agents is called Swarm Intelligence. Algorithmic models of such behavior are referred to as Computational Swarm Intelligence (Engelbrecht, 2005).

Examples of collective behavior in nature are numerous. They are based on direct or indirect exchange of information about the environment between the members of the swarm. Although the rules governing the interactions at the local level are usually easy to describe, the result of such a behavior is difficult to predict. However, through collaboration the swarms in nature are able to solve complex problems that are crucial for survival in a dynamically changing environment.

The process of formation of the complex patterns out of the multiplicity of interactions is referred to as emergence. The understanding and study of emergence in nature has itself been an emergent process, where a large number of researchers from various natural and social science disciplines have contributed to it. This process passed through several phases. It began with a few inquiring minds trying to understand the forces behind the emergent behavior without having solid scientific models to associate it with. In the second phase, scientists began to notice and compare the

(a) Ant colony (b) Bee swarm

(c) Bird flock (d) Fish school

Figure 2.1: Examples of swarming behavior in nature.

emergent behavior in different contexts which enabled them to understand the rules of conduct at the local level that lead to the emergence of global patterns. Finally, today, we are not only analyzing the emergent behavior, we started creating it.

2.2 Social Insects

Swarming behavior in nature has always intrigued scientists, and many studies have been aimed to its better understanding. The examples of such behavior are many. Ants communicate by laying pheromone on their route in order to find the shortest path to a food source. Termites cooperate to build complex nest structures without having any global knowledge of the environment. Bees use dancing to recruit other members in the swarm to follow them to the location in the field that is rich with nectar. Birds gather in flocks and fish form schools to have better chances of survival against predators. Bacteria use molecules to locally exchange the information about their environment, and so forth. Some examples of swarming behavior in nature are shown in Fig. 2.1.

Many aspects of swarming behavior have self-organizing properties. Self-organization is the ability of a group of agents, or in this case a swarm, to perform a task with-

out the need for central control. The rules specifying the interactions between the members of a swarm are executed on the basis of purely local information, without reference to the global pattern, which is an emergent property of the swarming behavior rather than a property imposed by an external ordering influence. Bonabeau et al. (1999) define four basic components of self-organization:

- **Positive feedback**: Examples are recruitment and reinforcement. For instance, recruitment to a food source is a positive feedback that relies on trail-laying and trail-following in some ant species, or dances in bees.
- **Negative feedback**: Counterbalances positive feedback and helps to stabilize the collective pattern in the form of saturation, exhaustion or competition. In the example of foraging, negative feedback stems from the limited number of available foragers, satiation, food resource exhaustion, crowding at the food source, or competition between food sources.
- **Randomness**: Examples are random walks, errors, random task-switching, and so on. Randomness is often crucial since it enables discovery of new solutions. For example, a lost ant can find a new source of food.
- **Multiple interactions**: A single individual can generate a self-organized structure (in ants, trail-following and trail-laying of pheromone can complement each other), but self-organization generally requires a minimal density of mutually tolerant individuals.

2.2.1 Pheromone Laying in Ant Colonies

Self-organizing properties of ant colonies were studied by Deneubourg et al. (1990). The authors showed how the ants mark their route with pheromone in order to attract other ants to follow them. Pheromone-following behavior helps the ants to find the shortest path to a food source, but it also helps them perform division of labor, allocation of resources or gathering of corpses. The distributed coordination of the ant colony was demonstrated in the binary bridge experiment depicted in Fig. 2.2. The experiment shows how over time the major part of the ant population chooses to take the shorter path from the nest to the food source.

In the experimental setup, the ants nest and the food source are connected by two branches of different length. Lets consider a simplified scenario where two ants leave the nest at the same time but take different branches to unknowingly reach the food source. Both ants lay pheromone on their path, but since the ant that took

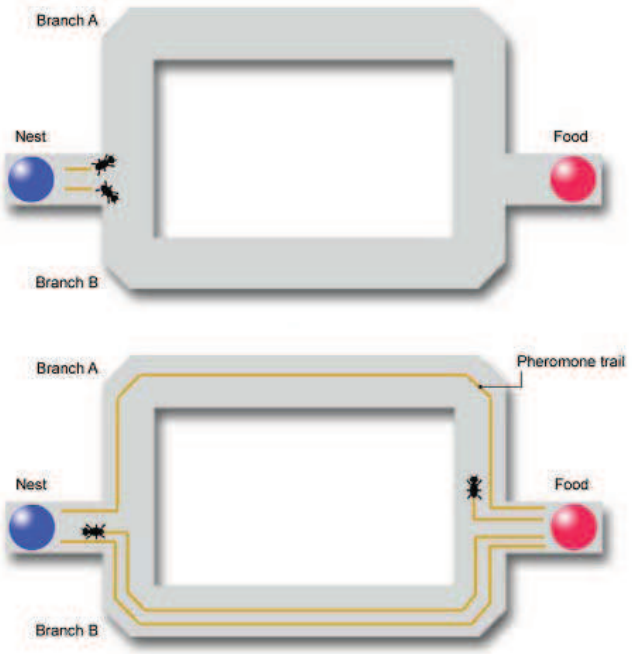


Figure 2.2: The binary bridge experiment: A higher concentration of accumulated pheromone on the shorter branch attracts more ants to follow this route from the nest to the food source.

the shorter branch is the first one to return to the nest, the pheromone trail on this branch will have a higher intensity level on a segment that is closer to the nest. If a third ant leaves the nest to start the search, it will detect the difference in intensity level of pheromone and choose the shorter branch. This ant will also lay pheromone on its path that lead to pheromone accumulation. By positive feedback more ants will be attracted to take the shortest path.

The model proposed by Deneubourg et al. assumes that the amount of pheromone on a branch is proportional to the number of ants that has crossed that branch. Let A_n and B_n be the number of ants that chose branches A and B, respectively, out of the first n ants. The probability that the $(n+1)$-st ant chooses the branch A is given by

$$p_{n+1} = \frac{(K + A_n)^\nu}{(K + A_n)^\nu + (K + B_n)^\nu} \tag{2.1}$$

where parameter K quantifies the degree of attraction of an unmarked branch and

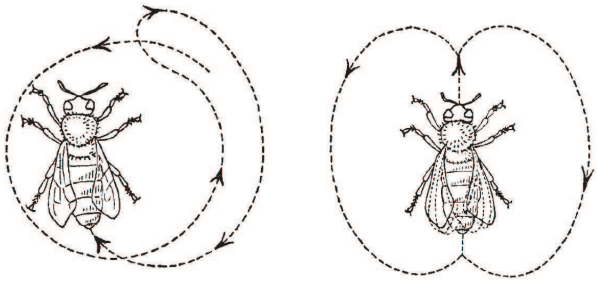
the value of ν determines the degree of nonlinearity of the choice function. ($K, \nu \geq 0$; $K, \nu \in \Re$.) For larger values of K, a greater amount of pheromone is needed to make the choice non-random. Also, when ν is large, one branch needs to have accumulated only slightly more pheromone than the other for the next ant to select it. The model by Deneubourg et al. was used as a basis for the group of algorithms called Ant Colony Optimization described in Section 3.2. An analysis of convergence of the proposed model is given in (Makowski, 2008).

2.2.2 Recruitment by Dance in Honey Bee Swarms

Honey bee dancing is one of the most intriguing behaviors in social insects. It is a form of direct communication that worker bees use to recruit other bees in the swarm to follow them to the resource site. The concept of dance in bees was described by Von Frisch (1967). The author demonstrated how the bees exchange the information about the distance and the direction of the food sources by using different movement patterns. Wenner and Wells (1990) suggest that bees communicate through floral odor present on their bodies upon return from a food source. Many experiments demonstrated the importance of floral odors in food location, yet the most commonly accepted view is that recruited bees go to the area depicted in the dance, but then home in on the flower patch using odor cues.

When a bee returns to the hive with a load of nectar that is sufficiently nutritious to guarantee return to the source, she performs a dance to share the information about the location of the food source with other bees. Dance patterns may contain two items of information, namely distance and direction of the food source. If a food source is located close to the hive (less than 50 m) the bee performs a round dance (Fig. 2.3(a)), but if the food source is at a greater distance (more than 150 m) the bee performs a "waggle dance" (Fig. 2.3(b)). For resource sites that are at intermediate distance, the bee performs a transitional sickle dance that contains elements of both the round dance and the waggle dance.

When performing a round dance, a forager bee communicates only the distance from the resource site and shares the load of nectar with the recruited bees. The waggle dance on the other hand contains the information of both distance and direction of the resource site. A bee that performs a waggle dance runs straight ahead for a short distance, returns in a semicircle to the starting point, runs again through the straight course, then makes a semicircle in the opposite direction to complete a full figure-eight circuit. While several variables of the waggle dance relate to distance, the duration of the straight-run portion of the dance is the simplest and most reliable



(a) Round dance (b) Waggle dance

Figure 2.3: Dance patterns of the bees. *(Image extracted from "Bees: Their Vision, Chemical Senses, and Language", K. Frisch, 1976)*

indicator. Their approximate linear relationship is shown in Fig. 2.4.

The direction, i.e. the angle, of the food source is indicated relative to the Sun. The angle that the bee adopts relative to the gravitational axis represents the angle of the site relative to the direction of the Sun. In other words, the dancing bee transposes the solar angle into the gravitational angle. A forager bee recruiting to a food source in the direction of the sun will perform a waggle dance with the straight-run portion directed upward. Conversely, if the food source is located away from the Sun, the

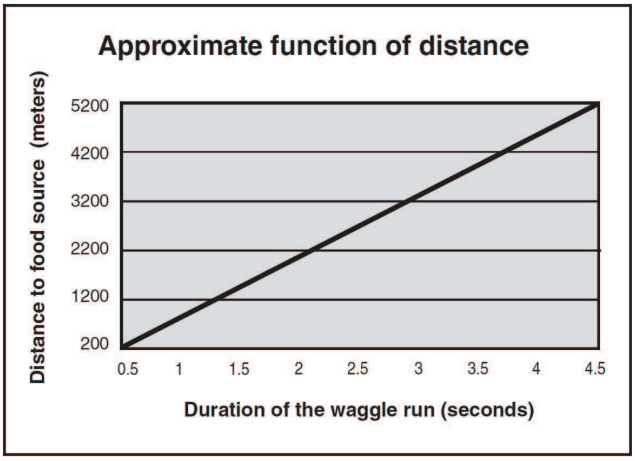


Figure 2.4: The relationship of distance to waggle dance straight-run duration. *(Image extracted from "The honey bee dance language", D. Tarpy)*

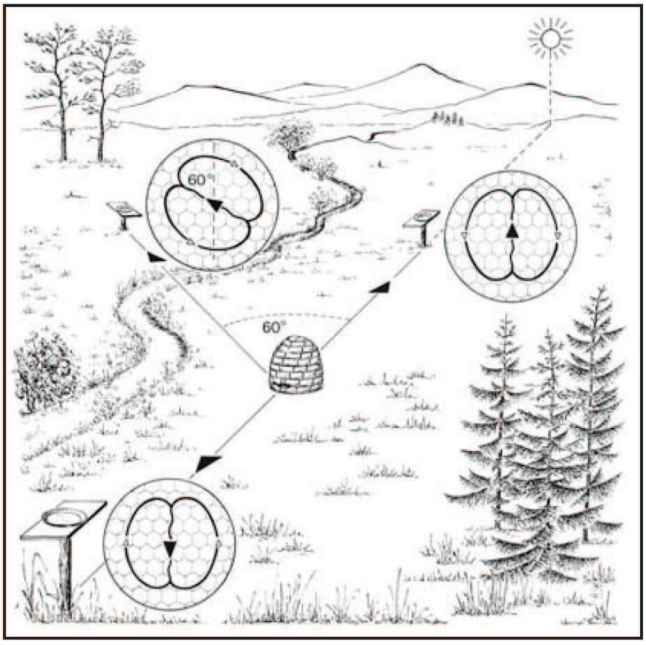


Figure 2.5: The orientation of the waggle dance compared to the resource site location. *(Image extracted from "Insects and Flowers: The Biology of a Partnership", F.G. Barth, 1982)*

straight run will be performed downward. This is shown in Fig. 2.5.

2.3 Flocking, Schooling and Aggregation

It could be noticed how the social organisms attain a survival advantage as the foraging efficiency is increased for groups rather than individuals. However, species can benefit from collective behavior in different ways. The capacity of all life-forms to develop complex behaviors without a central master planner has been a subject of study for decades. Turing (1952) described the hypothesis of pattern formation in flowers, focusing on the recurring numerical patterns. He demonstrated using mathematical tools how a complex organism assembles without any kind of centralized control.

One of the most interesting examples of self-organization in nature is found in slime mold. Keller and Segel (1970) showed how a swarm of slime mold cells aggregate to form a single organism in order to move to a more fertile area. When the food is abundant, the slime mold cells independently move around as individual amoebas

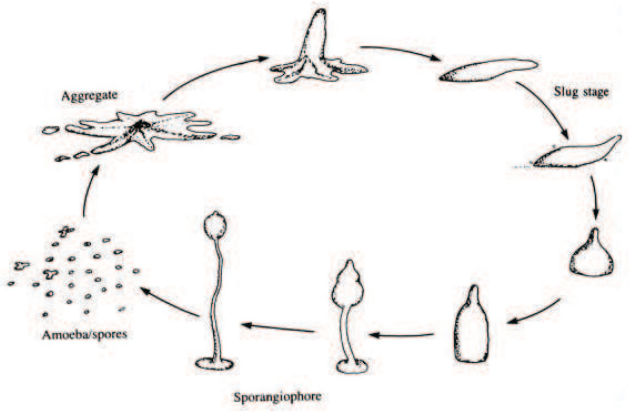


Figure 2.6: The aggregation of the slime-mold cells. *(Image extracted from Thomas Schmickl's website, University of Graz, Austria)*

throughout their substrate. But when the environmental situation changes they aggregate to a single multi-cellular body (see Figure 2.6). This is done by the means of local communication using a chemical signal called cAMP, emitted by cells to guide the collective movements. The cells follow the gradient of cAMP concentration which leads to their clustering. The slime mold aggregation mechanism was later validated through simulation by Resnick (1994).

Some social animals tend to organize and move in more orderly ways. Fish that form schools and birds that gather in flocks appear to move as one, as if they were guided by a leader. For a long time, this was the overall-accepted theory, but today we know that the individuals in such swarms sense the local environment to adjust their movement to that of their neighbors. Partridge (1982) showed that fish have the school-centering tendency that is achieved through visual contact with their neighbors in order to adjust the movement accordingly. Hamilton (1971) proposed that behaviors like schooling, flocking, and herding are seen more often in prey than predators, and are a result of an individual animal's attempt to seek protection away from the outer regions of the swarm where it could be picked up by a predator. Reynolds (1987) assumed that birds in flocks were driven by local collision avoidance, velocity matching, and flock centering. Additionally, Breder (1954) argued that the attraction of a school for a solitary fish depends on the number of the perceived fish and the distance between them.

2.4 Summary

This chapter introduced the concept of Swarm Intelligence and self-organized behavior in nature. Some animal societies, such as swarms of insects, are able to adapt to dynamically changing environment, which has been crucial for their survival. In particular, the examples of ant and bee colonies are pointed out, which served as inspiration for the algorithmic models presented in the following chapters.

Chapter 3

Computational Swarm Intelligence

3.1 Introduction

Computational Intelligence belongs to the broader field of Artificial Intelligence, and comprises of the paradigms that relate to some kind of biological or naturally occurring system (Andina and Pham, 2007). Computational Swarm Intelligence represents the group of Computational Intelligence algorithms that model swarming behaviors in nature. These algorithms are generally applied to optimization problems and other problems that can be converted to optimization problems.

Artificial swarms are usually designed using the bottom-up approach. The designer of this kind of systems needs to set the rules governing the mutual local interactions between the agents in the swarm with one another and between the agents and the environment. The indirect communication via environment is referred to as stigmergy. Although various definitions of the term stigmergy have been proposed (Shell and Matarić, 2003), in the context of swarm intelligence it is used to describe the exchange of information through modifications of the environment (Bonabeau et al., 1999). That is, an agent modifies the environment that in return modifies the behavior of other agents, i.e. they respond to it.

The initial purpose of the swarm-based algorithms was to solve optimization problems. However, in recent years, these algorithms have shown their full potential in terms of flexibility and autonomy when it comes to design and control of complex systems that consist of a large number of autonomous agents. In more general terms, these can be referred to as System of Systems (Jamshidi, 2009). What distinguishes

them is that they exploit the decentralization property of natural swarms in order to create autonomous, scalable, and adaptive multi-agent systems.

In this chapter, the state-of-the-art Computational Swarm Intelligence algorithms are introduced. First, the Ant Colony Optimization metaheuristic is described that served as basis for the algorithms proposed in Part II. Following, an overview of the algorithms that model the foraging behavior of bee colonies, which served as inspiration for the distributed task allocation algorithm proposed in Part III.

3.2 Ant Colony Optimization

Ant Colony Optimization (ACO) is a swarm-based metaheuristic which models the foraging behavior of ant colonies in nature (Dorigo and Stützle, 2004). As described in Chapter 2.2.1, the ants through collaboration can solve complex problems such as finding the shortest path to a food source. This feature can be used to solve the engineering problems that require this kind of optimization.

Artificial ants, unlike their biological counterparts, move through a discrete environment defined by nodes, and they have memory. When traversing from one node to another, ants leave pheromone trails on the edges connecting the nodes. The pheromone trails attract other ants that lay more pheromone, which consequently leads to pheromone trail accumulation. Negative feedback is applied through pheromone evaporation that, importantly, restrains the ants from taking the same route and allows continuous search for better solutions. Ant System (AS) is the first ACO algorithm proposed in literature and it was initially applied to the Traveling Salesman Problem (TSP) (Dorigo et al., 1996).

3.2.1 Traveling Salesman Problem

A general definition of the TSP is the following. For a given set of cities with known distances between them, the goal is to find the shortest tour that allows each city to be visited once and only once. In more formal terms, the goal is to find the Hamiltonian tour of minimal length on a fully connected graph.

Consider a set N of nodes, representing cities, and a set E of arcs (or edges) fully connecting the nodes N. Let d_{ij} be the length of the arc $(i, j) \in E$, that is, the distance between cities i and j, with $i, j \in N$. The TSP requires to find a minimal length Hamiltonian circuit on the graph $G = (N, E)$, where a Hamiltonian circuit of graph G is a closed tour visiting once and only once all the $n = |N|$ nodes of G, and its length is given by the sum of the lengths of all the arcs of which it is composed. Thus,

the optimal solution to the TSP is a permutation π of the node indices $1, 2, \ldots N$ such that the length $f(\pi)$ is minimal, and it is given by

$$f(\pi) = \sum_{i=1}^{N} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)} \tag{3.1}$$

The TSP is one of the most studied combinatorial optimization problems. A large number of different algorithmic techniques have been either applied to the TSP or developed to solve it. Early approaches include construction heuristics, iterative improvement algorithms, and exact methods like branch&bound or branch&cut (Lawler et al., 1985). Since the beginning of the 1980s, various metaheuristics have been tested on the TSP such as simulated annealing, tabu search, evolutionary algorithm, and iterated local search (Johnson and McGeoch, 1997). The TSP has served as a benchmark test for newly proposed combinatorial optimization algorithms and a collection of well-known solved problems can be found in the TSPLIB benchmark library which is accessible on the Web (Reinelt, 1991).

3.2.2 Ant System Algorithm

The AS algorithm generates a colony of artificial ants that move between the nodes (cities) in search for the optimal tour. The two main phases of the algorithm constitute the ants' solution construction and the pheromone update. The general ACO metaheuristic is shown in Algorithm 1. The optional local search phase was not considered for the AS algorithm, but it was applied in some extensions of the AS introduced in Subsection 3.2.3.

Algorithm 1 The Ant Colony Optimization Metaheuristic

```
Set parameters, initialize pheromone trails
while termination condition not met do
  ConstructAntSolutions
  ApplyLocalSearch (optional)
  UpdatePheromoneTrails
end while
```

At each construction step, ant k applies a probabilistic action choice rule, called *roulette* rule (or *wheel-selection* rule). That is, every node is associated to a probability

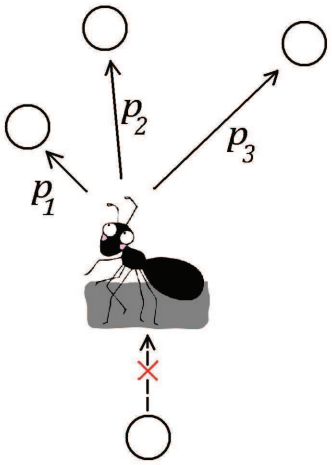


Figure 3.1: Ant displacement based on the nodes' associated probabilities.

of being chosen from a set of non-visited nodes, as graphically shown in Figure 3.1. The probability of displacing ant k from node i to node j is given by:

$$p_{ij}^{k} = \begin{cases} \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{h \notin tabu_k,} (\tau_{ih})^{\alpha}(\eta_{ih})^{\beta}} & \text{if } j \notin \text{tabu}_k \\ \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

where τ_{ij} and η_{ij} are the intensity of the pheromone trail on edge (i, j) and the visibility of the node j from node i, respectively, and α and β are the control parameters ($\alpha, \beta > 0$; $\alpha, \beta \in \Re$). The tabu_k list contains the nodes that have already been visited by the kth ant. The node's visibility is defined as inversely proportional to the node's distance:

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{3.3}$$

It can be concluded from the equations (3.2) and (3.3) that the ants favor the edges that are shorter and contain a higher concentration of pheromone.

AS is performed in iterations. At the end of each iteration the pheromone values are updated by all the ants that have built a solution in the iteration itself. The

pheromone update rule is described with the following equation:

$$\tau_{ij(new)} = (1-\rho)\tau_{ij(old)} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{3.4}$$

where ρ is the pheromone evaporation rate ($0 < \rho < 1$, $\rho \in \Re$), m is the number of ants in the colony, and $\Delta\tau_{ij}^{k}$ is the amount of pheromone laid on the edge (i, j) by the kth ant, and is given by:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k} & \text{if edge } (i,j) \text{ is traversed by the k}th \text{ ant} \\ \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

where L_k is the length of the tour found by the kth ant, and Q is a constant.

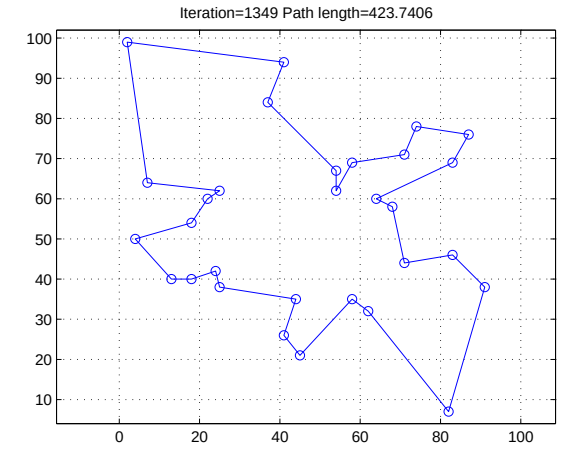
The algorithm stops when the satisfactory solution is found or when the maximum number of iterations is reached.

One configuration of the TSP is a 30-city problem "Oliver30-TSP" with the optimal tour length of 423.7406. The solution found after applying the AS algorithm is shown in Figure 3.2(a). It was obtained after 1349 iterations, which is shown on the solution convergence graph in Figure 3.2(b).

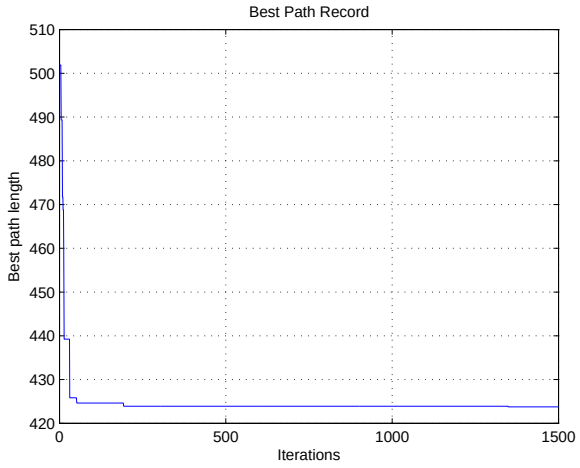
3.2.3 Extensions of Ant System Algorithm

Several ACO algorithms have been proposed in literature as extensions of the AS algorithm. They were introduced with certain modifications in order to achieve a better performance. The two most successful variants are the $\mathcal{MAX} - \mathcal{MIN}$ Ant System ($\mathcal{MM}$AS) and the Ant Colony System (ACS).

The $\mathcal{MM}$AS algorithm (Stützle and Hoos, 2000) introduced several modifications to the original AS. First, it exploits the best solutions found by depositing the pheromone only on the best tour in the current iteration or the global best tour. In order to prevent the algorithm stagnation that may come as a result of the pheromone accumulation on the preferred tours, the pheromone deposits are limited to a certain range defined by the lower and the upper limit values. This allows the constant exploration of new tours. The pheromone trails are initialized to the upper limit and the pheromone evaporation rate is set to a low value in order to increase the exploration of tours when the algorithm starts the search. Finally, if the algorithm approaches stagnation with no improvement of the best tour, the pheromone trails are re-initialized.



(a)



(b)

Figure 3.2: Ant System algorithm results on the Oliver30 TSP: a) optimal solution; b) solution convergence.

The ACS algorithm (Dorigo and Gambardella, 1997) introduced a local pheromone update in addition to the pheromone update performed at the end of the solution construction process. The local pheromone update is performed after each construction step by all the ants in the swarm. It is applied only to the last edge traversed by removing some pheromone from that edge to increase the exploration of the new

paths by subsequent ants in the same iteration. This encourages the ants to produce a number of different solutions during one iteration. Another modification to the AS algorithm is that, at the end of each iteration, the ACS performs pheromone evaporation and pheromone depositing only on the edges belonging to the best-so-far tour.

Some successors of the AS algorithm include Elitist AS (Dorigo et al., 1996), Ant-Q (Gambardella and Dorigo, 1995), Rank-based AS (Bullnheimer et al., 1999), ANTS (Maniezzo, 1999), BWAS (Cordon et al., 2000), and Hyper-cube AS (Blum and Dorigo, 2004), among others. One feature of the AS extensions is that they direct the search in a more aggressive way by giving emphasis to the best tour in each iteration (e.g., in $\mathcal{MM}$AS) or the best-so-far tour (e.g., in ACS). These produce a better solution quality for a faster computation time in the applications of the TSP. But for the applications described in Part II, such as the edge detection in images or the cluster analysis, the objective is not to find the best solution but to record the movement of all ants, which is reflected in the distribution of pheromone trails. For example, in case of image processing, we do not search for the strongest edge in the image, but for all the true edges that make the objects' boundaries. This is the rationale for using the AS algorithm as the basis for the proposed methods.

3.3 Bee Colony-inspired Algorithms

The foraging behavior of the bees described in Section 2.2.2 can be applied to optimization (find the best source of food), distributed task allocation (recruitment of the bees based on the quality of the food sources), etc. When a forager bee finds a food source, she returns to the hive and performs a "dance" in order to recruit other bees. The information about the richness and the location of the site is passed through direct communication on the central dance floor. Some models of the cooperative behavior of bee colonies with centralized communication or no communication have already been proposed in the literature.

Pham et al. (2006) developed the Bees Algorithm (BA), which in its basic version performs a random search combined with a neighborhood search and can be used for optimization. The algorithm exploits the concept of the central dance floor in order to select the fittest sites, but no direct communication between the swarm members exist. The recruitment of the bees can be done in a deterministic way according to the fitness values associated with the sites, or these fitness values can be used to determine the probability of the bees being selected. Together with scouting, this

differential recruitment is a key operation of the BA. The algorithm is performed in iterations, and it is stopped when the solution is found within the provided error margin or when the maximal number of iterations is reached.

Bailis et al. (2010) proposed a model of the bee colony foraging to investigate the value of sharing food source position information in different environments. The authors show through simulations that in environments of highly-scattered food, relying solely on private information about previously encountered food sources is more efficient than sharing information. However, in the nectar-rich environments it leads to decreased foraging efficiency.

Another bee colony-based optimization algorithm called Artificial Bee Colony (ABC) was proposed by Karaboga and Akay (2009). The concept of the central dance floor is applied through different roles that bees have in the swarm, namely scout, onlooker and employed bees. The algorithm uses recruitment based on the fitness values of the food sources and applies neighborhood search for solution improvements. A scout bee is randomly sent to search for a new food source when a previously found source is abandoned.

Few other algorithms inspired by bees' behavior appeared in literature, such as BeeHive (Wedde et al., 2004), BCO (Teodorović and Dell'Orco, 2005), Virtual Bee Algorithm (Yang, 2005), HBMO (Afshar et al., 2007), etc., and they have mostly been applied to solving combinatorial optimization problems.

3.4 Particle Swarm Optimization

The problem-solving behavior that emerges from the multiplicity of interactions between the individual agents is not common only to social insects. Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995) is a metaheuristic inspired by the flocking behavior of birds. In terms of this bird flocking analogy, a particle swarm optimizer consists of a number of particles, or birds, that fly around the space, or the sky, in search of the best location.

Each of these particles corresponds to a simple agent that moves through a multidimensional search space sampling an objective function at various positions. The motion of a given particle is dictated by its velocity which is continuously updated in order to pull it towards its own best position and the best positions experienced by the neighbors in the swarm. The performance of each particle is measured using a predefined fitness function which encapsulates the characteristics of the optimization problem. Kennedy et al. (2001) describe particle swarms as closely related to

cellular automata, which have three main attributes: (1) update of individual cells is performed in parallel, (2) each cell's new value depends only on its old value and the old value of the neighboring cells, and (3) all cells are updated using the same rules.

Many variants of the PSO algorithm have been proposed. Some of these intend to incorporate the capabilities of the evolutionary computation techniques, such as hybrid PSO (HPSO) (Naka et al., 2003), evolutionary PSO (EPSO) (Miranda and Fonseca, 2002), and differential evolution PSO (DEPSO) (Zhang and Xie, 2003). Different authors have suggested various adjustments to the parameters of the PSO algorithm by applying Fuzzy logic (Shi and Eberhart, 2001), inertia weight and constriction factors (Eberhart and Shi, 2000), or a secondary PSO algorithm (Doctor et al., 2004), among others. Most PSO algorithms, unlike ACO, are designed for search in continuous domains.

3.5 Advantages of Swarm Intelligence

The self-organizing properties of animal swarms have been studied for better understanding of the underlying concept of decentralized decision-making in nature. Disadvantages of decentralized control appear as conflicts between the members of swarms, redundant activities, and slow global response to a change in the environment. However, distributing the control of a swarm widely among its members almost certainly enhances its ability to make a rapid, local response to a change because it eliminates the need for time-consuming communication between its central and peripheral parts. More importantly, the bottom-up design topology applied to Swarm Intelligence algorithms provides them with higher autonomy, scalability, robustness and adaptability to the changes in their environment.

3.5.1 Scalability and Robustness

Scalability and robustness define the ability of a multi-agent system to scale-up or scale-down its performance with respect to the number of agents or number of tasks at hand. For a system comprised of a very large number of agents, a scalable design can significantly improve the performance of the overall system. On the other hand, a robust design insures that losing some agents will not cause a catastrophic failure.

3.5.2 Adaptation and Learning

The terms "adaptation" and "learning" are differently defined from perspective of different scientific studies. Psychologists define learning as a "modification of a behavioral tendency by experience". From the perspective of computer science, especially in robotics, learning is usually associated with Reinforcement Learning, or "how to map situations to actions so as to maximize a numerical reward signal". In our work, the term "adaptation" is used, which seems more appropriate and avoids confusion. It is used to describe the modifications in behavior of the system as whole, where the performance of a single agent on a specific task does not change over time.

The adaptation of animals swarms throughout the evolution allowed them to survive despite the dynamically changing environment. This feature is used to model the autonomous multi-agent systems or to create the adaptive tools for knowledge extraction and combinatorial optimization. For autonomous multi-agent systems, such as robot swarms, the ability to adapt to a dynamically changing environment is of utmost importance. The systems that are not adaptive are most likely to fail faced with unknown working conditions. On the other hand, the adaptive algorithms can be applied to any kind of dynamically changing digital habitat, such as real-time image processing or clustering of a variable data set.

3.6 Proposed Methodology

The proposed methodology consists of a set of simple rules that serve as guidelines for the design of the Swarm Intelligence tools, and they are as follows:

- Define the nodes that constitute the discrete data space in which the agents move.
- Define the set of application-specific variables and apply Equation 3.2 to calculate the probabilities of the displacement to the neighboring nodes.
- Apply the roulette rule as the underlying decision-making mechanism.
- Define the objective function for solutions evaluation.

3.7 Summary

This chapter gives an overview of the state-of-the-art Computational Swarm Intelligence algorithms. Scalability, robustness and adaptation are described as their dis-

tinct features. The artificial swarms are used to simulate natural swarms, to model swarms of physical agents (e.g. robots), or their problem-solving behavior can be applied to optimization problems, which was their initial purpose. This cooperative behavior shows emergent properties and produces unpredictable global patterns. One way of dealing with the unpredictability issue is statistical analysis.

In order to facilitate the design of the methods based on Swarm Intelligence, a general methodology is proposed that consists of a set of simple rules. In the following chapters, the main contributions of this thesis are presented as case studies. In Part II, the general design methodology is validated in a real-world scenario and novel Swarm Intelligence methods are proposed as tools for optimization and feature extraction. In Part III a swarm-based model of a multi-agent system is presented for distributed task allocation in a swarm of robots.

Part II

Tools for Optimization and Feature Extraction

Chapter 4

Route Optimization of Unmanned Aerial Vehicles

4.1 Introduction

Ant Colony Optimization (ACO) is a metaheuristic method initially proposed to solve combinatorial optimization problems. Ant System (AS) is the first ACO algorithm proposed in literature and it was initially applied to the Traveling Salesman Problem (TSP) (Dorigo et al., 1996), a well-known NP-hard optimization problem already described in Subsection 3.2.1. The concept of the TSP has many applications. In digital communications, when one deals with the large networks of communicating agents such as the servers on the Internet, the objective of finding the shortest possible communication route has become even more critical. Nevertheless, future applications will include a wide range of software and hardware agents that need to communicate or displace in an optimal manner. One of such examples is Unmanned Aerial Vehicles (UAVs).

UAV is an aircraft without the onboard presence of pilots. It was initially used for military operations, but the interest for its involvement in commercial applications is on the rise. These include telecommunications, ground traffic control, search and rescue operations, and crop monitoring among others. In September 2002, NASA's solar-powered Pathfinder-Plus UAV was used to conduct a proof-of-concept mission in U.S. national airspace above a 3500 acre commercial coffee plantation in Hawaii. UAVs assist with frost protection, irrigation and crop management in agriculture. Together with Mobile Ground Station systems, UAVs offer persistent surveillance, enhanced situational awareness, and actionable intelligence to law enforcement and

security personnel on the move.

UK law enforcement have studied the use of small VTOL UAVs with a stills camera, daylight TV sensor and a live video downlink, for urban surveillance and crowds. California-based AeroVironment's UAV can stay aloft for a week at 65.000 ft, providing low-cost communications relays and aerial mapping. The KB4 is used to track icebergs. The plane is able to fly in swarms of three, collaborating autonomously on some in-flight decisions. An inverted-V tail helps keep Aerosonde stable in high winds making it ideal for hurricane monitoring. The 5-pound SkySeer can be used for police search-and-rescue missions, as well as scouting forest fires and counting migratory animals.

UAVs have several basic advantages over manned systems including increased maneuverability, reduced cost, reduced radar signatures, longer endurance, and less risk to crews. One of the challenges in the control of UAVs is to make them autonomous or semi-autonomous in order to relieve the operator from the constant monitoring. One application is the area coverage, where the task is to find the minimal route that connects a defined set of waypoints. This can be treated as a TSP, and various soft-computing methods have been successfully applied in order to solve it.

In this chapter, the application of the AS algorithm to the UAVs' route optimization is described. The UAVs are engaged in a simulated area coverage scenario with a defined set of waypoints. The objective is to find the shortest route that connects all the waypoints in order to optimize the time and the cost of the UAV's flight. The effectiveness of the AS algorithm is shown in comparison with the Nearest Neighbor Search (NNS) algorithm which was initially used in the UAV scenario simulation (Jaimes and Jamshidi, 2010).

The chapter is organized as follows. Section 4.2 gives an overview of the related work. A detailed description of the AS algorithm is given in Section 4.3. The problem statement and the proposed method are described in Section 4.4. In Section 4.5 the experimental setup and the discussion of the results are presented. Finally, in Section 4.6 the conclusions are made.

4.2 Related Work

4.2.1 Soft-computing Methods for Traveling Salesman Problem

The TSP has been studied intensively and it is often used as a benchmark for the new optimization algorithms. Many exact and metaheuristic algorithms have been applied to the TSP. The exact algorithms include tour construction algorithms such

as the well-known Nearest Neighbor Search (NNS) (Arya et al., 1998; Rosenkrantz et al., 2009). The NNS algorithm is fast and easy to implement but often produces near-optimal solutions that are not satisfactory, especially for a large number of cities.

The metaheuristic algorithms use imprecision and approximation to find the optimal solution. These algorithms have been applied successfully to the TSP by a number of researchers. Some widely applied algorithms are Simulated Annealing (SA) (Bonomi and Lutton, 1984; Golden and Skiscim, 1986), Tabu Search (TS) (Knox, 1994) and Genetic Algorithms (GA) (Grefenstette et al., 1985; Whitley et al., 1989; Nguyen et al., 2007).

The ant colony metaphor is easily adapted to the TSP because of the inherent feature of the ants foraging behavior to search for the shortest path to the food source. The AS algorithm was first proposed by (Dorigo et al., 1996) as a multi-agent approach to solving the TSP. Other variants of the AS algorithm have been proposed since, most successful being the Ant Colony System (ACS) (Dorigo and Gambardella, 1997) and the *MAX-MIN* Ant System (MMAS) (Stützle and Hoos, 2000).

4.2.2 Ant Colony Optimization for Unmanned Aerial Vehicle Applications

The ACO-based algorithms have been used for the UAV applications. Ma et al. (2007) proposed an ant colony-based method for the UAV global optimal trajectory planning. The obtained optimal route was not a feasible UAV trajectory, so the authors applied a trajectory smoothing method. Another method was proposed by Zhenhua et al. (2008) who used Voronoi diagrams to create a set of possible trajectories and then applied the Multiobjective Ant Colony System algorithm to find the optimal route. Duan et al. (2009) applied the pheromone-laying pheromone-following behavior to the team of UAVs for collision avoidance and simultaneous arrival to the target in dynamic and uncertain environments.

4.3 Ant System Algorithm

Artificial ants, unlike their biological counterparts, move through a discrete environment defined with nodes, and they have memory. When moving from one node to another, ants leave pheromone trails on the edges connecting the nodes. The pheromone attracts other ants, which creates a positive feedback that leads to a pheromone trail accumulation. A negative feedback is applied through pheromone evaporation that,

importantly, restrains the ants from taking the same route, therefore prevents the algorithm stagnation.

After defining the discrete environment in which the artificial ants can move, the AS algorithm starts with an initialization step which is followed by iterative construction of new solutions and pheromone update. The AS algorithm involves the following steps:

1. Initialization: The population of ants is created by placing one ant on every node. The edges are assigned with an initial pheromone trail, τ_0.

2. Node transition rule: Ants are allowed to displace to any node they have not already visited. The list $tabu_k$ contains all the nodes visited by the kth ant. Every node has an associated probability with which it is chosen from the set of available nodes. This decision-making mechanism is known as the *roulette rule*. The probability of displacing the kth ant from the node i to the node j is given by:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{h \notin tabu_k} \tau_{ih}^\alpha \eta_{ih}^\beta}, & \text{if } j \notin tabu_k \\ \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

where τ_{ij} and η_{ij} are the intensity of the pheromone trail on the edge (i, j) and the visibility of the node j from the node i, respectively. ($\tau_{ij}, \eta_{ij} > 0$; $\tau_{ij}, \eta_{ij} \in \Re$; for $\forall i, j$.) The parameters, α and β, are used respectively to bias the search in favor of the exploitation of the accumulated pheromone or exploration of new solutions, respectively. ($\alpha, \beta > 0$; $\alpha, \beta \in \Re$.) $Tabu_k$ list contains the nodes visited by the kth ant. The visibility of the node j from the node i is defined as the reciprocal value of their Euclidean distance, d_{ij},

$$\eta_{ij} = \frac{1}{d_{ij}}. \tag{4.2}$$

Once all the probabilities are calculated as in (4.1), ant will choose a node by "spinning the roulette wheel". This step is repeated until all the nodes are visited by all the ants. The best solution (the shortest path found) from this iteration is compared with the overall best and the minimum of the two is saved as the new overall best.

3. Pheromone update rule: When all the ants have finished their route, the pheromone update is applied. The edges that form a part of some ant's route accumulate more pheromone, and those that don't lose pheromone trails through evaporation. This is given by:

$$\tau_{ij(new)} = (1-\rho)\tau_{ij(old)} + \Delta\tau_{ij} \tag{4.3}$$

where ρ is the pheromone evaporation rate ($0 < \rho < 1$; $\rho \in \Re$), and $\Delta\tau_{ij}^k$ is the amount of pheromone laid on the edge (i, j) by the kth ant, and is given by:

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{4.4}$$

where

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k}, & \text{if edge } (i,j) \text{ belongs to the route} \\ \\ 0, & \text{otherwise.} \end{cases} \tag{4.5}$$

where L_k is the kth ant's route length and Q is a constant.

4. Stopping criterion: The steps 2 and 3 are repeated in the loop and the algorithm stops executing when the maximum number of iterations is reached.

4.4 Problem Statement and Proposed Method

4.4.1 Traveling Salesman Problem

In the proposed scenario, the team of UAVs is sent to fly over a certain number of locations on the ground. This problem can be interpreted as a well-known NP-hard optimization problem called the Traveling Salesman Problem (TSP). A general definition of the TSP is the following. Consider a set N of nodes, representing cities, and a set E of arcs fully connecting the nodes N. Let d_{ij} be the length of the arc $(i, j) \in E$, that is, the distance between nodes i and j, with $i, j \in N$. The TSP is the problem of finding a minimal length Hamiltonian circuit on the graph $G = (N, E)$, where a Hamiltonian circuit of the graph G is a closed tour visiting once and only once all the $n = |N|$ nodes of G, and its length is given by the sum of the lengths of all the arcs of which it is composed.

Finding the shortest tour means saving time for task execution, as well as saving energy needed for the UAVs' flight. The UAVs start their tour from a base which they constantly communicate with by sending the information of their position and direction of flight (roll, pitch, and yaw). The calculation of the optimal tour is performed in the base and sent to the UAVs as a list of coordinates that need to be visited. Each tour starts from the location in the list that is closest to the base. By taking into account the starting position of the UAVs, maximum savings in time and energy are obtained for the task at hand.

4.4.2 Proposed Method

The proposed method (Jevtić et al., 2010a) is based on the foraging behavior of ant colonies where, in search for food, ants leave pheromone trails in order to attract other ants to follow their routes. For the UAV route optimization problem, the waypoints, i.e. their coordinates, represent the nodes. The edges between the nodes are the aerial paths UAVs take to move from one location to another not taking into account the dynamics of the flight. More precise optimization results would be obtained if the angle of turns the UAVs make would be calculated. Still, considering that the distance between the nodes is large enough, the best calculated route remains the same.

The input for the proposed method is the list of the waypoints' coordinates that need to be visited by the UAV. The AS algorithm is an iterative process and includes the steps mentioned in Section 4.3. The number of ants equals the number of nodes, and each node is a starting point of a different ant. Each edge is initiated with the same value of the pheromone trail, τ_0, so that the initial probabilities that edges would be chosen depend on their length only. This helps the ants find satisfactory good solutions in the first iteration.

The ants move from node to node based on the node transition rule in (4.1) until they visit all the nodes exactly once. The distance from the last node to the starting node is added to the total tour length. Each iteration ants produce solutions and the best tour is saved as the iteration's best. This is compared with the best solution from the previous iterations in order to find the global best.

The pheromone trails on the edges visited by ants are updated as in (4.2) and (4.3), which leads to pheromone accumulation. Unvisited edges lose pheromone by evaporation and become less attractive to the ants in the subsequent iterations. The algorithm stops executing when the maximum number of iterations is reached. The global best is the optimal tour found for the given list of waypoints.

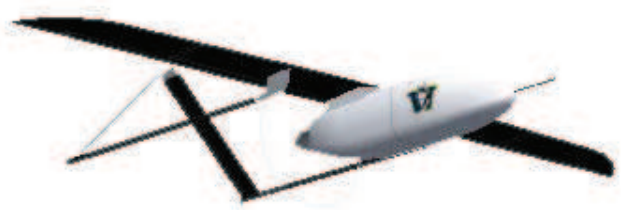


Figure 4.1: Aerosonde UAV: length 5 ft 8 in (1.7 m), height 2 ft 0 in (0.6 m), wingspan 9 ft 8 in (2.9 m), wing area 6.1 ft^2 (0.57 m^2).

4.5 Experimental Results

4.5.1 UAV Simulation

The simulation of the UAV's flight was done using the MATLAB/SIMULINK (software MATLAB, version R2009b) in conjunction with the Aerosim library. The library contains all the necessary blocks to simulate different airplane models. It also comes with an Aerosonde UAV dynamic model preloaded (see Fig. 4.1). The UAV is controlled by a decentralized fuzzy logic control. Each control houses three rules with multiple stages (Gomez and Jamshidi, 2010).

4.5.2 Communication Protocol

The communication protocol can be categorized in two types. The protocol interested only in the transport of data and the protocol interested in the actual mining of the data. The first protocol has both electrical and protocol definitions on how the data is transmitted, and is known as the transport layer. The protocol concerned in the actual mining of the data is also called application protocol or application layer. An example of these two protocols is the transmission of HTML protocol accomplished by another protocol such as TCP.

For the application layer, a protocol based on a Modbus message structure has been created. Modbus is an application layer protocol based on client/server architecture. Usually, it presents two serial modes: RTU and ASCII. The ASCII serial frame, used in our protocol, is represented in Fig. 4.2.

Figure 4.2: Modbus ASCII serial frame.

4.5.3 Simulation of Waypoint Navigation

In order to test the communication protocol, a MATLAB simulation has been performed to simulate waypoint navigation. The simulation has been performed using two computers. The first hosts the user interface for the ground station (see Fig. 4.3). The second hosts the user interface for the airplane's model simulation (see Fig. 4.7). The model was created using the Aerosim in Simulink.

Both computers are communicating to each other using the Xstream radio modem. The main objective of this simulation is to test the communication algorithm. That includes the algorithms to build and send the different messages and to read and process the received messages. The block diagram of the communication algorithm run in the ground station to monitor the location of the UAVs is shown in Fig. 4.4. The algorithm to assign the new waypoints to the UAVs is shown in Fig. 4.5.

4.5.4 Results and Discussion

Initially, for a small number of waypoints, the optimal UAV's route was obtained using the Nearest Neighbor Search (NNS) algorithm (Arya et al., 1998). Although the NNS was often able to compute the optimal solution, it was very dependent on the waypoints distribution. One simple example of the TSP for which the NNS algorithm

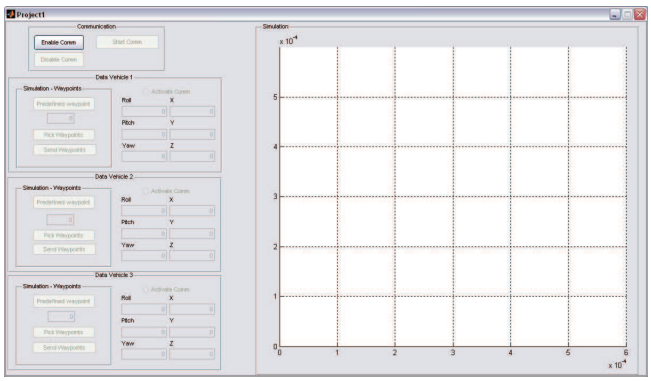


Figure 4.3: Ground station user interface for the waypoint navigation.

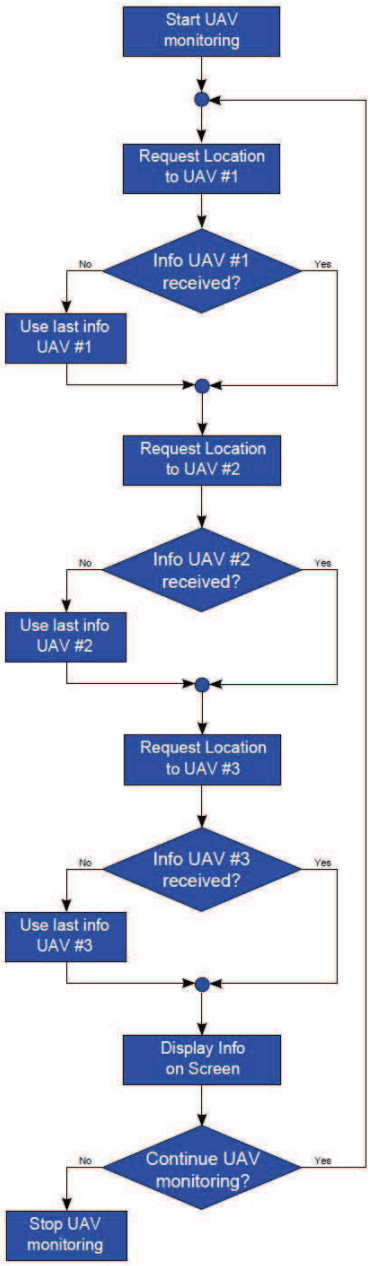


Figure 4.4: The block diagrams of the communication algorithm for monitoring the UAVs location.

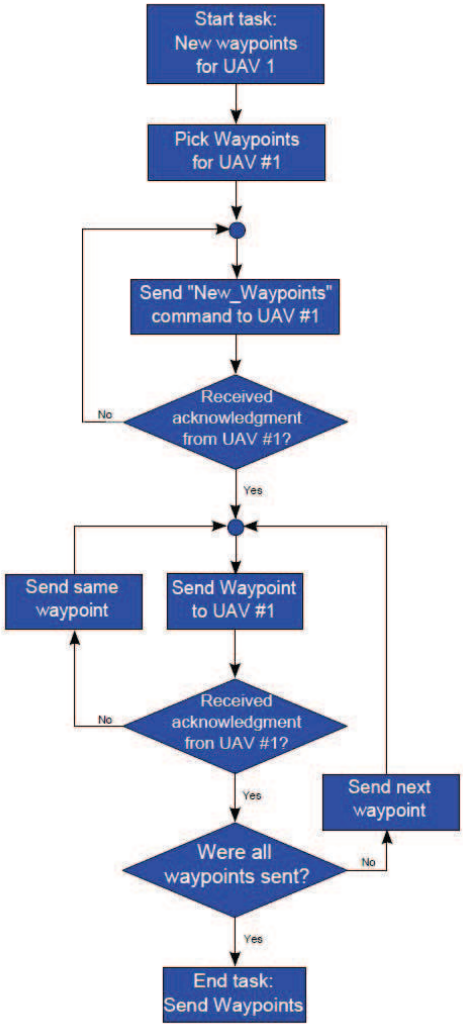
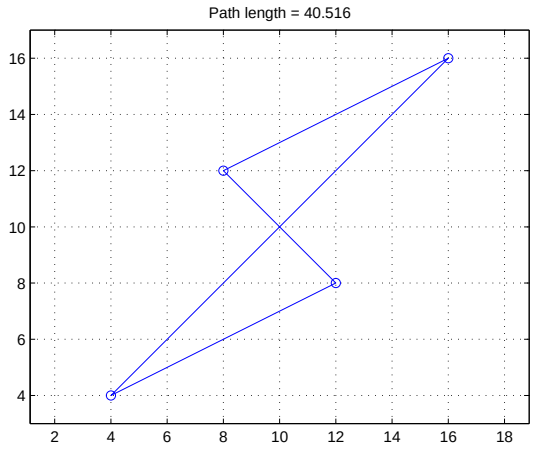


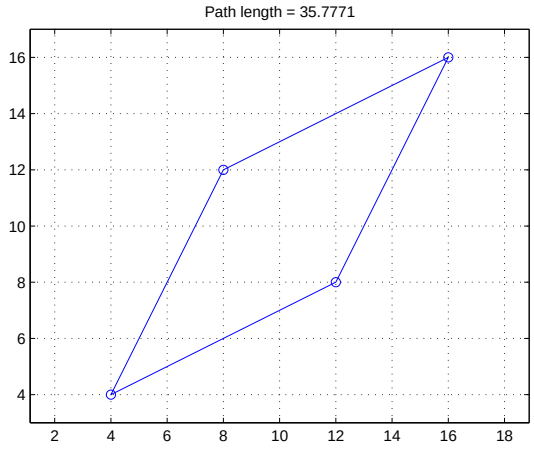
Figure 4.5: The block diagram of the communication algorithm for assigning the new waypoints to the UAVs.

is not able to compute the optimal solution is shown in Fig. 4.6.

In our experiments, the Ant System (AS) algorithm was used to calculate the optimal route in the base and send the arranged list of waypoints to UAVs to perform the flight. The comparison of the results of NNS and AS algorithms is shown in Table 4.1. The number of waypoints shown in the first column was randomly distributed



(a) Nearest Neighbor Search algorithm



(b) Ant System algorithm

Figure 4.6: Solution comparison of the optimal route for a 4-city Traveling Salesman Problem obtained by the NNS and the AS algorithms.

(except for the Ulysses16 TSP and Oliver30 TSP problems), and for each problem 100 experiments were performed. The "AS vs. NNS" column shows the result improvement by the AS algorithm with respect to the NNS algorithm. The last column shows the number of iterations used for the AS algorithm.

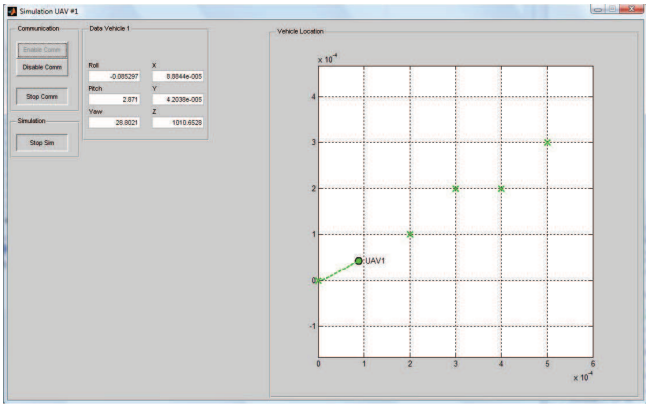
The number of waypoints used in the experiments was limited to 30 (Oliver30

Table 4.1: Comparison of the Results for the NNS and AS algorithms

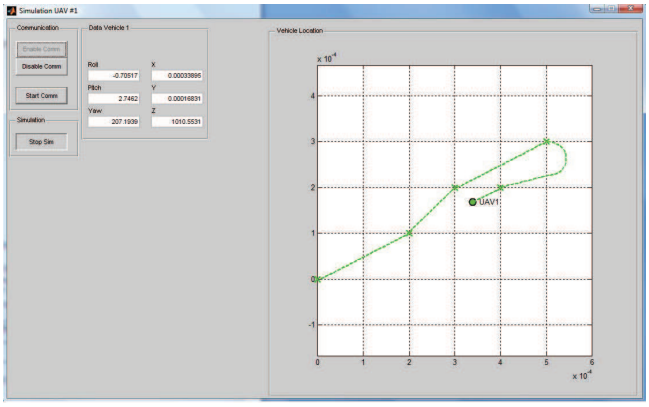
No. of waypoints	Distance type	NNS best	AS best	AS average	Improvement by AS [%]	No. of iter.
4	euclid	99.89	99.89	99.89	0.00	400
5	euclid	158.88	158.88	158.88	0.00	500
6	euclid	181.41	181.41	181.41	0.00	600
7	euclid	295.36	280.83	280.83	4.92	700
8	euclid	271.98	262.03	262.03	3.66	800
9	euclid	244.77	239.63	239.63	2.10	900
10	euclid	349.32	325.85	325.85	6.72	1000
11	euclid	308.87	294.49	294.49	4.66	1100
12	euclid	325.19	323.41	323.41	0.55	1200
13	euclid	339.33	315.84	315.84	6.92	1300
14	euclid	385.24	378.92	378.92	1.64	1400
15	euclid	397.75	357.22	357.22	10.19	1500
16 (Ulysses16)	geo	7835.00	6747.00	6747.00	13.89	3000
30 (Oliver30)	euclid	473.33	423.74	423.76	10.48	3000

TSP) due to limited hardware resources for the UAV's simulation, and also in practical terms of UAV's applications. The results in Table 4.1 show that AS outperforms NNS, especially for larger sets of waypoints. Since the solution obtained by NNS depends on the waypoints' distribution that was random in the experiments, the solution improvement by the AS is not constant (in percentage) but it is notable and is rising for the scenarios with larger numbers of waypoints.

The parameters α, β, ρ and Q directly or indirectly affect the probability defined in (4.1). The parameters α and β represent the relative importance of the pheromone trail on the edge and the visibility of the node, respectively. By changing their values we bias the ants to exploit the knowledge contained in the accumulated pheromone trails or explore new solutions. The parameter ρ is the pheromone evaporation rate that prevents the algorithm stagnation, which would be a result of the amplification of the initial solution. The constant Q is related to the pheromone trail laid by the ants after constructing a solution. The set of values used in the experiments is based on the experimental setup described in (Dorigo et al., 1996): $\alpha = 1$, $\beta = 5$, $\rho = 0.5$ and $Q = 100$.



(a) Simulation start



(b) Simulation end

Figure 4.7: UAV flight simulation for four waypoints.

The number of iterations for AS was proportional to the number of waypoints (multiplied by 100). This value was empirically obtained, and it can be seen from the results in Table 4.1 that the average AS results are equal to the best AS results, which means that the ant colony was able to find the optimal solution in every experiment (except for the Oliver30 TSP).

In Fig. 4.7 the user interface frames from the start and the end of the UAV's flight simulation are shown. The 4-waypoint set was used for which the NNS algorithm does not give the optimal solution. It can be noticed that the UAV's route is not a straight line connecting the waypoints, which comes as a result of the dynamic model of the vehicle.

4.6 Summary

In this chapter, the Ant System algorithm was applied to the route optimization of the UAVs in the simulated area coverage scenario. The experimental results show that the algorithm outperformed the Nearest Neighbor Search algorithm that was initially used to find the optimal route. The interest for the usage of the UAVs in commercial applications is on the rise and the method proposed in this chapter is an improvement in terms of time and energy consumption for the UAV's flight. This can be of great importance in applications where a large number of UAVs is used, or when a prompt action is needed and the resources are scarce.

The problem of the UAV's route optimization served as a testbed for the Swarm Intelligence-based optimization tool and the proposed design methodology. It is also an illustrative example that will help understand the underlying principles of the novel tools for feature extraction and multi-agent system modeling proposed in the following chapters.

Chapter 5

Adaptive Edge Detection in Digital Images

5.1 Introduction

Edge detection is a pre-processing step in applications of image segmentation and computer vision. It transforms the input image to a binary image that indicates either the presence or the absence of an edge. Therefore, the edge detectors represent a special group of search algorithms with the objective of finding the pixels belonging to true edges. The search is performed following certain criteria, as the edge pixels are found in regions of an image where the distinct intensity changes or discontinuities occur (e.g. in color, gray-intensity level, texture, etc.).

The purpose of edge detection is to segment the image in order to extract the features or the regions of interest. No matter what method is applied, the objective remains the same, to change the representation of the original image into something easier to analyze. But digital images can be obtained under different lighting conditions and by applying different techniques, which may produce noise and deteriorate the segmentation results. Various edge detectors apply different approaches to address these issues.

In recent years, the algorithms based on the swarming behavior of animal colonies in nature have been applied to edge detection. Swarm-based algorithms use the bottom-up approach where the patterns that appear at the system level are the result of the local interactions between its lower-level components. Although the initial purpose of these algorithms was to solve optimization problems, they proved to be a useful image processing tool. The inherent emerging properties of the, so called,

Swarm Intelligence approach make these algorithms adaptive to changes in digital image habitat. This can serve as a useful feature when it comes to real-time image processing.

In this chapter, two edge-detection algorithms inspired by the foraging behavior of natural ant colonies are presented. The ants use pheromone trails to mark the path to the food source. In case of digital images, pixels define the discrete space in which the artificial ants move, and the pixels belonging to true edges represent the food. The edge detection operation is performed on a set of grayscale images. The first proposed algorithm extracts the edges from the original grayscale image, while the other is applied to finding the missing parts of the broken edges and can be used as a complementary tool to any of the existing edge detectors. Finally, the study on the adaptability of the ant-based edge detector is performed where a set of grayscale images is used to create a variable environment.

The chapter is organized as follows. Section 5.2 provides an overview of the state-of-the-art edge detectors. In Section 5.3 the proposed Ant System-based edge detector is described. The discussion of the simulation results is also given in this section. Follows the description of the proposed broken-edge linking algorithm in Section 5.4. The simulation results are also presented in this section. The study on the adaptability of the proposed Ant System-based edge detector is given in Section 5.5. Finally, in Section 5.6 the conclusions are made.

5.2 Related Work

Edges represent important contour features in the image since they are the boundaries where distinct intensity changes or discontinuities occur. In practice, it is difficult to design an edge detector capable of finding all the true edges in image. The edge detectors give ambiguous information about the location of object boundaries for which they are usually subjectively evaluated by the observers (Shin et al., 2001).

Various edge detection methods have been proposed in literature. The Prewitt operator (Prewitt, 1970) was proposed to extract contour features by fitting a Least Squares Error (LSE) quadratic surface over a 3×3 image window and differentiate the fitted surface. The edge detectors proposed by Sobel and Feldman (1968) or Canny (1986) use local gradient operators, sometimes with additional smoothing for noise removal. The Laplacian operator (Gonzalez and Woods, 2008) uses a second order differential operator to find edge points based on the zero crossing properties of the processed edge points.

Although conventional edge detectors usually perform linear filtering operations, there are various nonlinear methods proposed. Panetta et al. (2008) proposed an edge detection method based on the Parameterized Logarithmic Image Processing (PLIP) and a four directional Sobel detector, achieving a higher level of independence from scene illumination. He et al. (2006) presented an edge detector based on bilateral filtering which achieves better performance than single Gaussian filtering. Mertzios and Tsirikolias (2001) proposed using the Coordinate Logic Filters (CLF) in order to extract the edges from images. CLF constitute a class of nonlinear digital filters that are based on the execution of Coordinate Logic Operations (CLO). Danahy et al. (2007) introduced an alternative method for calculating CLF using Coordinate Logic Transforms (CLT) and presented a new measure and thresholding technique for the detection of edges in grayscale images.

ACO algorithms have also been applied to image processing. Some of the proposed applications include image retrieval (Ramos et al., 2002) and image segmentation (Huang et al., 2008; Khajehpour et al., 2005). Several ACO-based edge detection methods have also been proposed in literature. Among others, these include modifications to Ant System (AS) (Nezamabadi-pour et al., 2006) or Ant Colony Systems (ACS) algorithms (Tian et al., 2008) for a digital image habitat, combined with local gray-intensity comparison for different pixel's neighborhood matrices.

5.3 Ant System-based Edge Detector

In this section, the AS-based edge detector proposed in (Jevtić et al., 2009b) is presented. The method requires that a set of images is extracted from the original grayscale image using a nonlinear image enhancement technique called Multiscale Adaptive Gain (Laine et al., 1994), and then the modified AS algorithm is applied to detect the edges on each of the extracted images. The result is a set of pheromone-trail matrices which are summed to produce the output image. Threshold and edge thinning are finally applied to obtain a binary edge image. The block diagram of the proposed method is shown in Fig. 5.1.

5.3.1 Multiscale Adaptive Gain

Image enhancement techniques emphasize important features in the image while reducing the noise. Multiscale Adaptive Gain is applied to obtain contrast enhancement by suppressing the pixels with the gray-intensity values of very small amplitude and

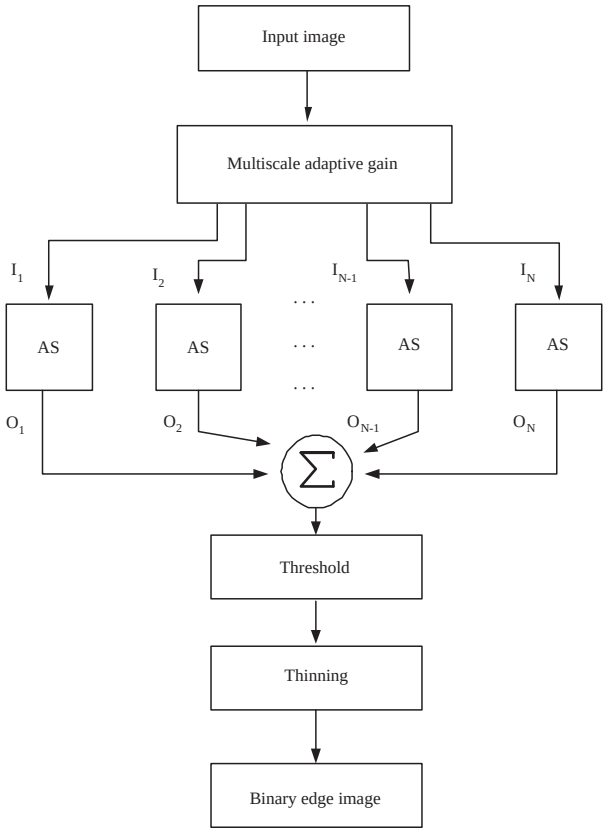


Figure 5.1: Block diagram of the proposed edge-detection method

enhancing the pixels with values larger than a certain threshold within each level of the transform space. The nonlinear operation is described with the following equation:

$$G(I) = A[sigm(k(I - B)) - sigm(-k(I + B))] \tag{5.1}$$

where

$$A = \frac{1}{sigm(k(1 - B)) - sigm(-k(1 + B))} \tag{5.2}$$

where $I = I(i, j)$ is the gray-intensity value of the pixel at (i, j) of the input image and $sigm(x)$ is defined as

$$sigm(x) = \frac{1}{1 + e^{-x}} \tag{5.3}$$

and B and k control the threshold and rate of enhancement, respectively. ($0 < B < 1$, $B \in \Re$; $k \in \aleph$). The transformation function (5.1) relative to the original image pixel values is shown in Fig. 5.2. It can be noticed that $G(I)$ is continuous and monotonically increasing, therefore, the enhancement will not introduce new discontinuities into the reconstructed image.

5.3.2 Ant System Algorithm for Edge Detection

The generic Ant System algorithm described in Section 3.2.2 was used as a base for the proposed edge detector. In digital images, the discrete environment in which the ants can move is defined by pixels, i.e. their gray-intensity values, $0 \leq I(i,j) \leq I_{max}$, $i = 1,2,\ldots,N$; $j = 1,2,\ldots,M$. Possible ant's moves to the neighboring pixels are shown in Fig. 5.3.

Unlike the cities' visibility in the TSP, the visibility of the pixel at (i,j) is defined as follows:

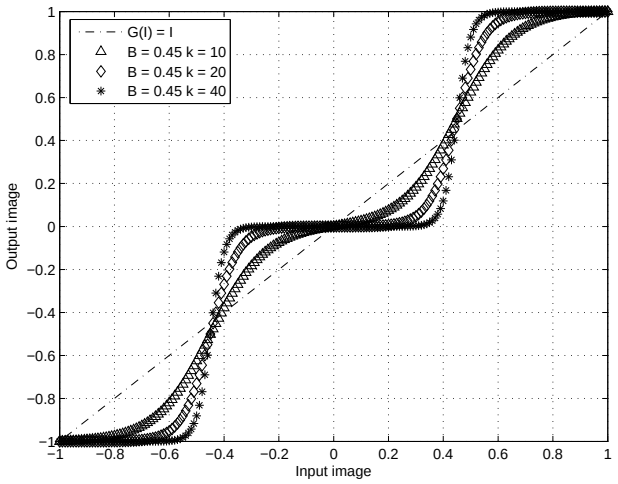
$$\eta_{ij} = \frac{1}{I_{max}} \cdot max \begin{bmatrix} |I(i-1,j-1) - I(i+1,j+1)|, \\ |I(i-1,j+1) - I(i+1,j-1)|, \\ |I(i,j-1) - I(i,j+1)|, \\ |I(i-1,j) - I(i+1,j)| \end{bmatrix} \tag{5.4}$$

where I_{max} is the maximum gray-intensity value in the image ($0 \leq I_{max} \leq 255$). For the pixels in regions of distinct gray-intensity changes the higher visibility values are obtained, which makes those pixels more attractive to ants.

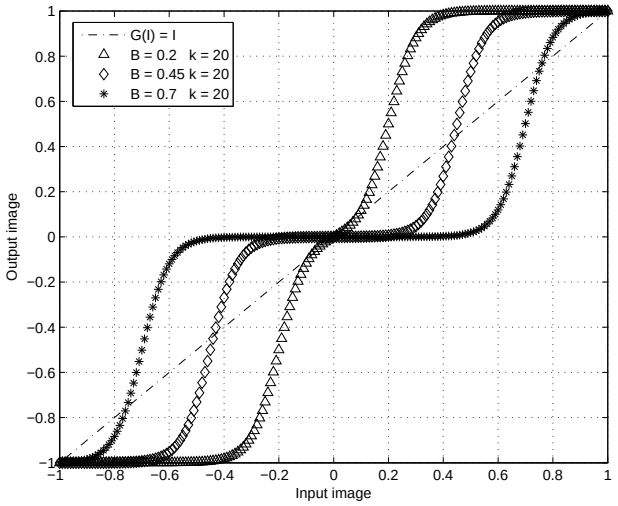
The AS algorithm is an iterative process which includes the following steps:

1. Initialization: the number of ants proportional to $\sqrt{N \cdot M}$ is randomly distributed on the pixels in the image. Only one ant is allowed to reside on a pixel within the same iteration. Initial non-zero pheromone trail value, τ_0, is assigned to each pixel, otherwise the ants would never start the search.

2. Pixel transition rule: Unlike their biological counterparts, artificial ants have memory. Tabu_k represents the list of pixels that the kth ant has already visited. If an ant is surrounded by the pixels that are either in the tabu list or occupied by other ants, it is randomly displaced to another unoccupied pixel that is not



(a)



(b)

Figure 5.2: Transformation function $G(I)$ in respect to the original image pixel values: (a) $B = 0.45$; $k = 10$, 20 and 40; (b) $B = 0.2$, 0.45 and 0.7; $k = 20$.

found in the tabu list. Otherwise, the probability for the kth ant to move to a

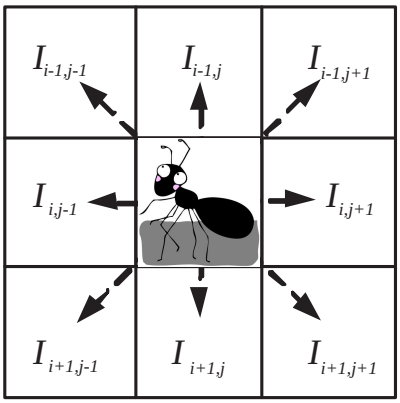


Figure 5.3: Proposed pixel transition model. *(Image extracted from (Nezamabadi-pour et al., 2006))*

neighboring pixel (i, j) is given by:

$$p^k_{(i,j)} = \begin{cases} \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_u \sum_v (\tau_{uv})^\alpha (\eta_{uv})^\beta}, & (i,j) \text{ and } (u,v) \text{ are allowed nodes} \\ 0, & \text{otherwise} \end{cases} \tag{5.5}$$

where τ_{ij} and η_{ij} are the intensity of the pheromone trail and the visibility of the pixel at (i, j), respectively, and α and β are control parameters ($\alpha, \beta > 0$; $\alpha, \beta \in \Re$).

3. Pheromone update rule: Negative feedback is demonstrated through the pheromone trails evaporation according to:

$$\tau_{ij(new)} = (1 - \rho)\tau_{ij(old)} + \Delta\tau_{ij} \tag{5.6}$$

where

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau^k_{ij} \tag{5.7}$$

and

$$\Delta\tau^k_{ij} = \begin{cases} \eta_{ij}, & \text{if } \eta_{ij} \geq T \text{ and } k\text{th ant displaces to pixel } (i,j) \\ 0, & \text{otherwise.} \end{cases} \tag{5.8}$$

T is a threshold value which prevents ants from staying on the background pixels hence enforcing the search for the true edges. By introducing the pheromone evaporation rate, ρ, the algorithm stagnation is prevented. In case of repeatedly not-visited pixels pheromone trail evaporates exponentially.

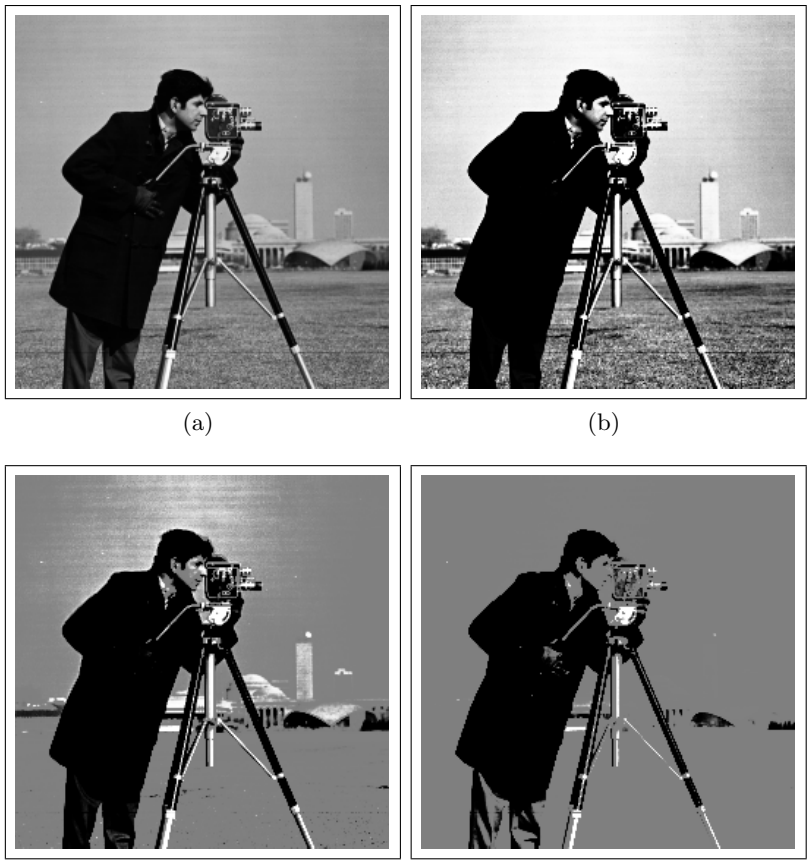
4. Stopping criterion: The steps 2 and 3 are repeated in a loop and algorithm stops executing when the maximum number of iterations is reached.

5.3.3 Simulation Results and Discussion

The proposed method was tested on four different grayscale images of 256×256 pixels resolution: "Cameraman", "Lena", "House" and "Peppers". As seen from the block diagram in Fig. 5.1, first the Multiscale Adaptive Gain defined in (5.1) is applied to the input image: $0 \leq I(i,j) \leq I_{max}$, $i = 1,2,\ldots,N$; $j = 1,2,\ldots,M$. ($N = M = 256$.) The values of B and k were varied to obtain a set of nine enhanced images: $B = 0.2$, 0.45 and 0.7; $k = 10$, 20 and 40. The effects of the transformation function on the image "Cameraman" are shown in Fig. 5.4. It can be noticed that, by changing the transformation function's parameters, some features in the image become highlighted while others get attenuated.

Afterwards, the AS-based edge detector is applied to each of the nine enhanced images. The algorithm's parameters are set as proposed in (Nezamabadi-pour et al., 2006): $\tau_0 = 0.0001$, $\alpha = 2.5$, $\beta = 2$, $\rho = 0.04$ and $T = 0.08$. The number of ants equal to $\sqrt{N \cdot M} = 256$ was randomly distributed over the pixels in the image with the condition that no two ants were placed on the same pixel. The memory length for each ant was set to be 10. The algorithm was stopped after 300 iterations generating a pheromone-trail matrix of the same resolution as the original image. After each of the nine enhanced images was processed, the sum of the pheromone-trail matrices produced the final pheromone-trail image (see Fig. 5.5(e)–(h)). A global threshold was applied to remove the irrelevant edges, i.e. the pixels with a lower accumulation of pheromone trails. Finally, by applying morphological edge-thinning (Pratt, 1991) the resulting binary image was obtained (see Fig. 5.5(i)–(l)).

The effectiveness of the proposed method was compared with the ant-based edge detectors proposed by Tian et al. (2008) and Nezamabadi-pour et al. (2006), and the results are shown in Fig. 5.6. To provide a fair comparison, the threshold and morphological edge-thinning operations are neglected, since they are performed as a post-processing step to further refine the edge information that is extracted by ACO. As seen from the figure, the proposed approach outperforms the other two



(a) (b)

(c) (d)

Figure 5.4: Effects of the transformation function $G(I)$; "Cameraman", 256×256 pixels: (a) original image; (b) $B = 0.2$, $k = 10$; (c) $B = 0.45$, $k = 20$; (d) $B = 0.7$, $k = 40$.

methods in terms of visual quality of the extracted edge information and sensitivity to weaker edges. The main contribution of the proposed edge-detection method is the preprocessing step and the parallel execution of the Ant System-based edge detector on a set of images that finally produce the output edge image. The execution time of the proposed method is high, which requires additional algorithm's code optimization and different programming environment. The presented experiments were performed in Matlab software, which offers an easy high-level programming experience, but is ineffective in terms of speed.

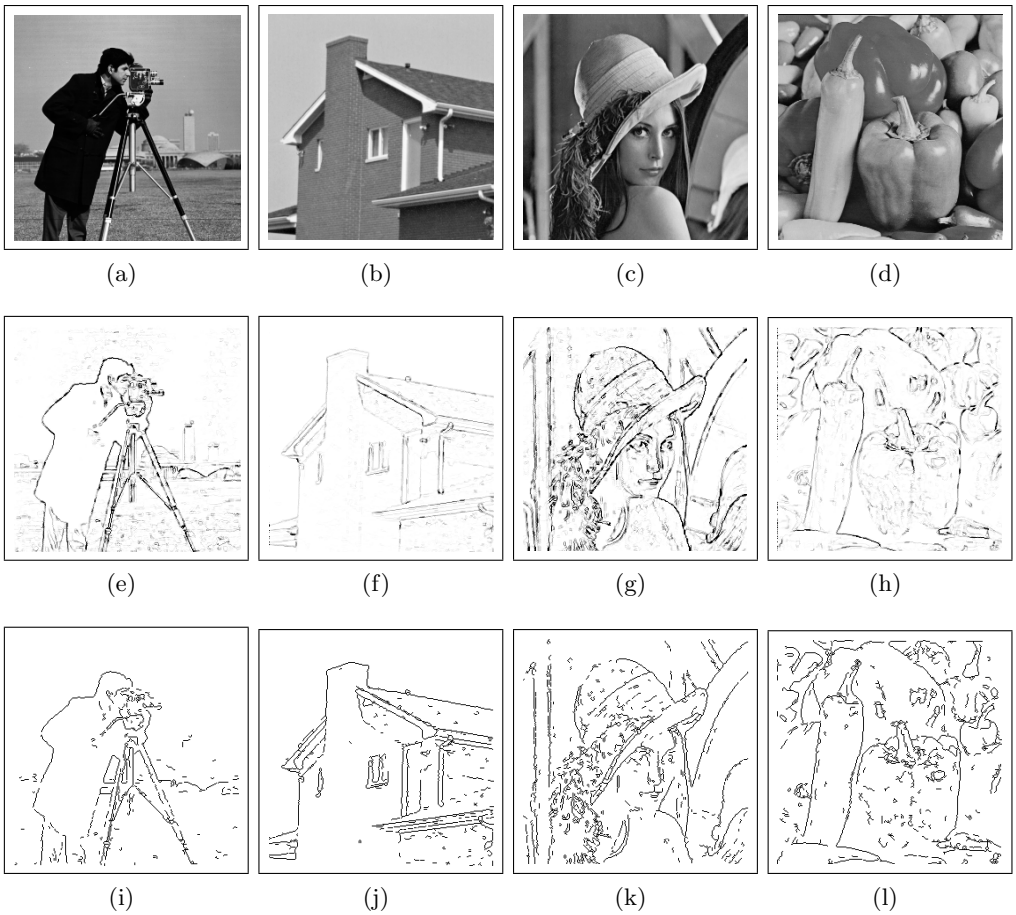
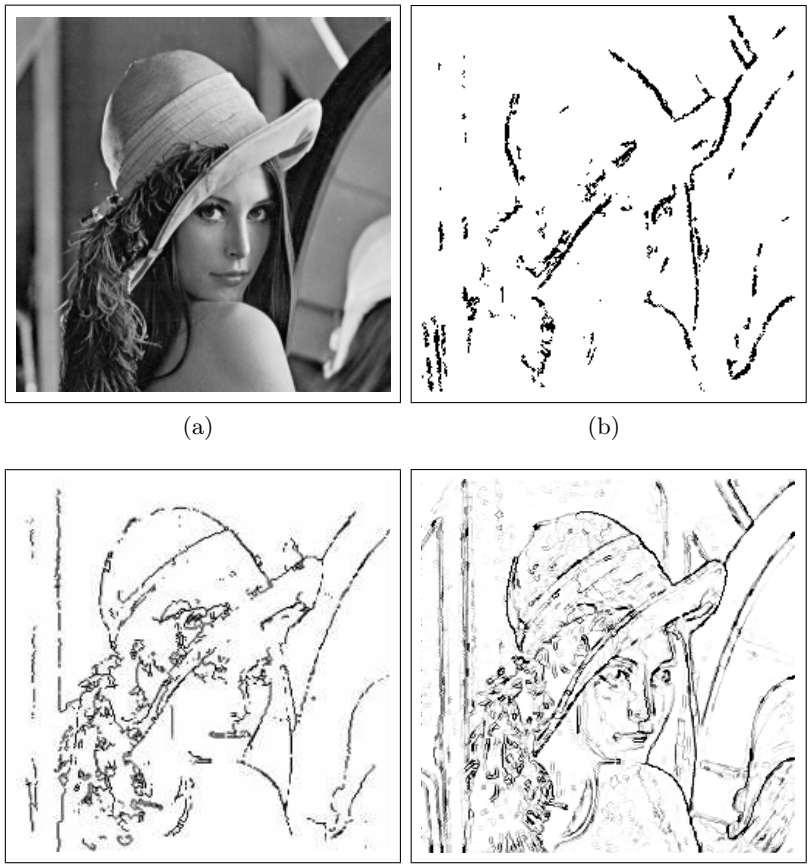


Figure 5.5: Qualitative results of the proposed method, with 256 × 256 pixel images: (a) "Cameraman" original image; (b) "House" original image; (c) "Lena" original image; (d) "Peppers" original image; (e) "Cameraman" pheromone trail image; (f) "Cameraman" edge image; (g) "House" pheromone trail image; (h) "Lena" pheromone trail image; (i) "Peppers" pheromone trail image; (j) "House" edge image; (k) "Lena" edge image; (l) "Peppers" edge image.

5.4 Ant System-based Broken-edge Linking Algorithm

Conventional image edge detection always results in missing edge segments. Broken-edge linking is an improvement technique that is complementary to edge detection. It is used to connect the broken edges in order to form the closed contours that separate



(a) (b)

(c) (d)

Figure 5.6: Comparative results with other ant-based edge detectors, "Lena" 256×256 pixels: (a) original image; (b) Tian *et al.*; (c) Nezamabadi-pour *et al.*; (d) the proposed method.

the regions of interest. The detection of the missing edge segments is a challenging task. A missing segment is sought between two endpoints where the edge is broken. The noise that is present in the original image may limit the performance of the edge-linking algorithms.

Many broken-edge linking methods have been proposed to compensate the edges that are not fully connected by the conventional edge detectors. Jiang et al. (2007) applied morphological image enhancement techniques to detect and preserve thin-edge features in the low contrast regions of an image. Wei et al. (2008) applied the

Sequential Edge-Linking (SEL) algorithm that provided full connectivity of the edges but for a rather simplified two-region edge-detection problem. Authors proposed this method to extract the contour of a breast as the region of interest in mammogram. Shih and Cheng (2004) applied adaptive structuring elements to dilate the broken edges along their slope directions. Lu and Chen (2008) proposed improvement to the traditional Ant Colony Optimization (ACO) based method for broken-edge linking to reduce the computational cost.

In this section, an Ant System-based broken-edge linking algorithm proposed in (Jevtić et al., 2009a) is presented. As inputs are used: the Sobel edge image and the original grayscale image. The Sobel edge image is a binary image obtained after applying the Sobel edge detector (Sobel and Feldman, 1968) to the original grayscale image. From this image the endpoints are extracted that will be used afterwards as the starting pixels for the ants' routes.

The original image is used to produce the *grayscale visibility* matrix, which for the pixel at (i, j) is calculated as follows:

$$\xi_{ij} = \frac{1}{I_{max}} \cdot max \begin{bmatrix} |I(i-1, j-1) - I(i+1, j+1)|, \\ |I(i-1, j+1) - I(i+1, j-1)|, \\ |I(i, j-1) - I(i, j+1)|, \\ |I(i-1, j) - I(i+1, j)| \end{bmatrix} \tag{5.9}$$

where I_{max} is the maximum gray value in the image, so ξ_{ij} is normalized ($0 \leq \xi_{ij} \leq 1$). For the pixels in regions of distinct gray intensity changes the higher values are obtained. The matrix of *grayscale visibility* will be the initial pheromone trail matrix. It is also used to calculate the fitness value of a route chosen by ant. The resulting image will contain the routes (connecting edges) with the highest fitness values found as optimal routes between the endpoints. In order to discard non-optimal routes, a fitness threshold is applied. Finally, the output image is the improved image that is a sum of the Sobel edge image and the connecting edges. The block diagram of the proposed method is shown in Fig. 5.7.

The proposed AS-based algorithm for broken-edge linking includes the following steps:

1. Initialization: The number of ants equals the number of endpoints found in the Sobel edge image, and each endpoint will be a starting pixel of a different ant. Initial pheromone trail for each pixel is set to its *grayscale visibility* value.

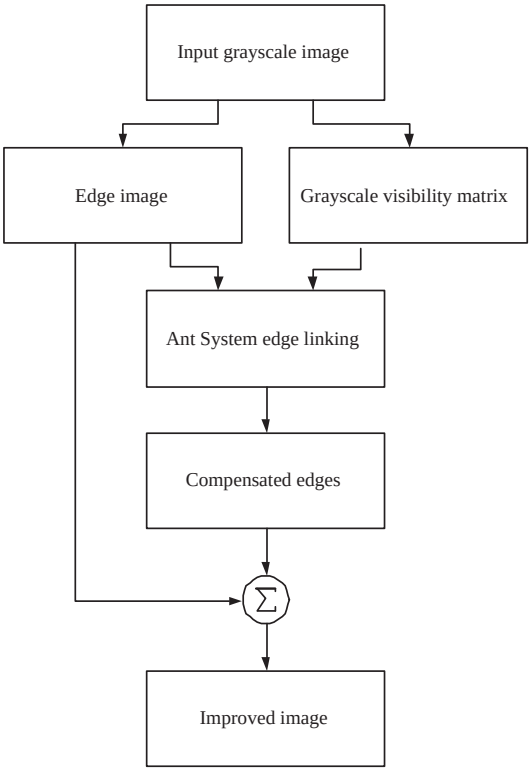


Figure 5.7: Block diagram of the proposed edge linking method

2. Pixel transition rule: Possible ant's transitions to the neighboring pixels are defined by the 8-connection pixel transition model shown in Fig. 5.3. The admissible neighboring pixels for the move of the kth ant are the ones not in the tabu$_k$ list. The probability for the kth ant to move from pixel (r, s) to pixel (i, j) is calculated as follows:

$$p^k_{(r,s)(i,j)} = \begin{cases} \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_u \sum_v (\tau_{uv})^\alpha (\eta_{uv})^\beta} & \text{if } (i,j) \text{ and } (u,v) \notin \text{tabu}_k, \\ & r-1 \le i,u \le r+1, \\ & s-1 \le j,v \le s+1 \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

where τ_{ij} and η_{ij} are the intensity of the pheromone trail and the visibility of the pixel at (i, j), respectively, and α and β are control parameters ($\alpha, \beta > 0$;

$\alpha, \beta \in \Re$). The visibility of a pixel should not be misinterpreted as its *grayscale visibility*, and for the pixel at (i, j) it is defined as:

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{5.11}$$

where d_{ij} is the Euclidean distance of the pixel at (i, j) from the closest endpoint.

3. Pheromone update rule: Negative feedback is created through the pheromone trails evaporation according to:

$$\tau_{ij(new)} = (1 - \rho)\tau_{ij(old)} + \Delta\tau_{ij} \tag{5.12}$$

where ρ is the pheromone evaporation rate ($0 < \rho < 1$; $\rho \in \Re$), and

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{5.13}$$

where

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{f_k}{Q} & \text{if } k\text{th ant displaces to pixel } (i, j) \\ \\ 0 & \text{otherwise.} \end{cases} \tag{5.14}$$

The fitness value of a pixel, f_k, is equal to the fitness value of the route it belongs to, and it is defined by the following equation:

$$f_k = \frac{\bar{\xi}}{\sigma_\xi \cdot N_p} \tag{5.15}$$

where $\bar{\xi}$ and σ_ξ are the mean value and the standard deviation of the *grayscale visibility* of the pixels in the route, and N_p is the total number of pixels belonging to that route. Pheromone evaporation prevents algorithm stagnation. From the repeatedly not-visited pixels the pheromone trail evaporates exponentially.

4. Stopping criterion: The steps 2 and 3 are repeated in a loop and algorithm stops executing when the maximum number of iterations is reached. An iteration ends when all the ants finish the search for the endpoints, by either finding one or getting stuck and being unable to advance to any adjacent pixel.

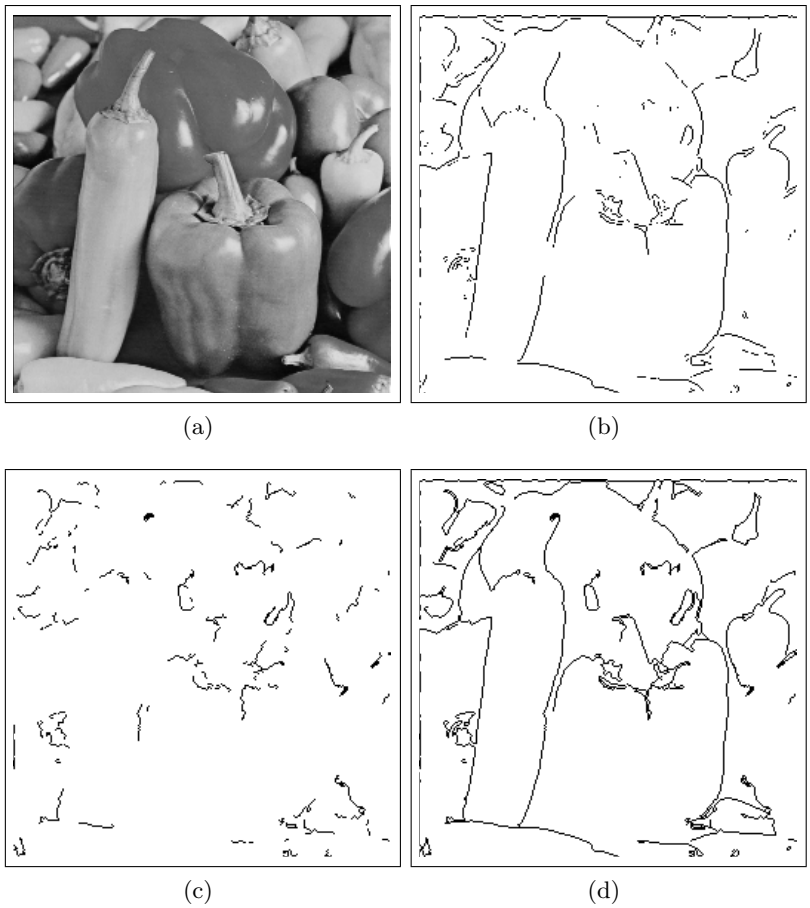


Figure 5.8: Qualitative results of the proposed edge-linking method, "Peppers" 256×256 pixels: (a) original image; (b) Sobel edge image; (c) resulting image of the proposed method; (d) improved edge image.

5.4.1 Simulation Results and Discussion

The simulation results of the proposed algorithm applied to the "Peppers" image of 256×256 pixels are shown in Fig. 5.8. The algorithm successfully detects the missing edge segments (Fig. 5.8(c)), as the self-organizing nature of ant colony optimizes the routes defined by edge pixels.

The initial pheromone trail for each pixel was set to its *grayscale visibility* value. In this manner, the pixels belonging to true edges have a higher probability of being chosen by ants on their initial routes, which shortens the time needed to find a satis-

factory solution, or improves the solution found for a fixed number of iterations. This allowed to obtain the results presented in just 100 iterations.

Designated values $\alpha = 10$ and $\beta = 1$ were determined on trial and error basis. A large α/β ratio forces the ants to choose the strongest edges. The existence of the control parameter β is important since it inclines the ant's route towards the closest endpoint. Experimental results showed that, by setting the β value to zero, it took more steps for the ants to find the endpoints which made the computation time longer. In some cases, ants were not even able to find the satisfactory solution for a reasonable number of steps, or they just got stuck between already visited pixels.

The effect of the α/β parameter ratio on the resulting image is best presented in Fig. 5.9. It can be noticed that the endpoint in the upper-left corner of the ROI image was not connected to any of the closer endpoints, and that the ants successfully found the more remote endpoint which was the correct one. The existence of the β parameter keeps the ants away from the low-contrast regions, such as the region of low gray-intensity pixels between two closer endpoints.

The ant's memory, i.e. the length of the tabu list, was set to 10. Larger ant's memory values would improve the quality of the resulting binary image but would as well lead to the prolonged computation time. The designated value was large enough to keep the ants from being stuck in small pixel circles.

The fitness value of a route is dependent on the mean value and the standard deviation of the grayscale visibility of the pixels in the route, and the total number of pixels belonging to that route, as defined in (5.15). The routes that have higher grayscale visibility mean value are the stronger edges as the gray level contrast of their adjacent pixels is higher. The smaller standard deviation of the grayscale visibility of the pixels in the route results in a higher fitness value. By this, more importance is given to the routes consisted of pixels belonging to the same edge, thus avoiding the ants crossing between the edges and leaving pheromone trails on non-edge pixels. Finally, the shorter routes are more favorable as a solution, therefore by keeping the total number of pixels in the route smaller, the higher fitness values are obtained.

The number of iterations was set to 100, which gave satisfactory results within an acceptable computational time of execution. The lower resolution images, for example 128×128 pixels, allowed larger number of iterations to be used, since a smaller number of ants was processed for a smaller number of relatively closer endpoints. The execution time of the algorithm was not optimal, and it was measured in minutes. One of the reasons is that the algorithm code was not written in an optimal manner since this was not a goal of the presented research work. Additionally, the Matlab as

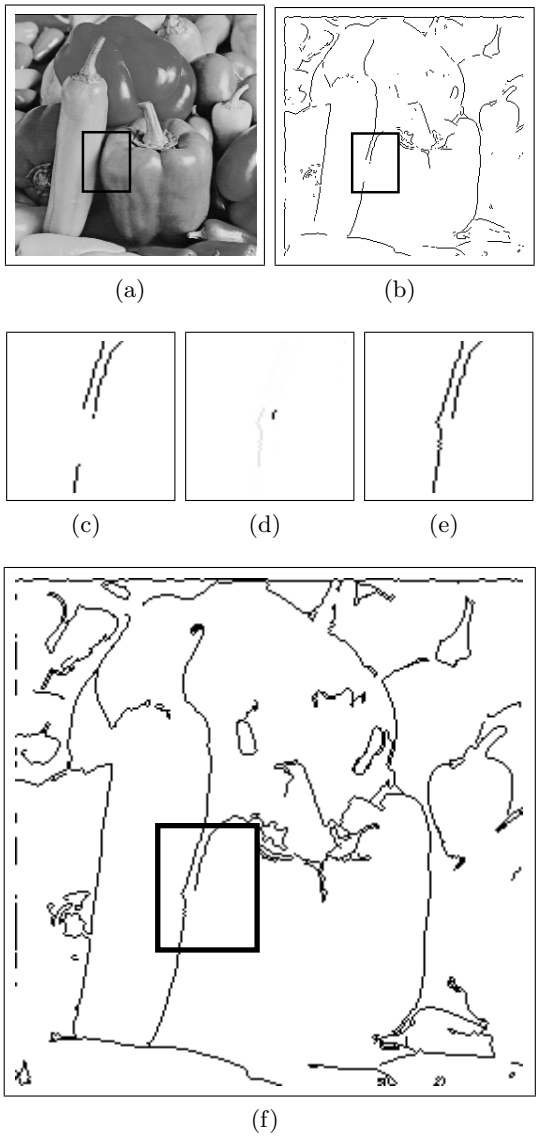


Figure 5.9: Effect of the control parameters on correct connection of the endpoints: Peppers, 256×256 image: (a) original image with marked region of interest (ROI); (b) Sobel edge image with marked ROI; (c) enlarged ROI: Sobel edge image; (d) enlarged ROI: pheromone trails image; (e) enlarged ROI: improved edge image; (f) improved edge image with marked ROI.

a programming environment is not intended for a fast code execution, rather for an easy high-level programming experience.

In order to test the proposed method on different input images, simulations were performed on "House", "Lena" and "Cameraman" images of size 256×256 pixels. The results confirm the effectiveness of the method, as shown in Fig. 5.10. It can be noticed that the found edge segments are often not unidirectional, which indicates that the fitness function was adequately defined and the ants found the true edges.

The main contribution of the proposed broken-edge-linking method is that it offers a different approach to solving this problem, which is based on the emergent behavior of the artificial ant colony. The comparison with the state-of-the-art methods was not presented, since the edge detection solutions are in most of the cases subjectively evaluated by the observer.

5.5 Adaptability of the Artificial Ant Colony

In recent years, Swarm Intelligence algorithms have shown their full potential in terms of flexibility and autonomy, especially concerning the design and control of complex systems that consist of a large number of agents. These algorithms demonstrate emergent behavior that results from the interactions of their lower-level components. They tend to be decentralized, self-organized, autonomous and adaptive to the changes in their environment. The adaptability and the ability to learn are very important for systems that are designed to be autonomous.

The learning ability, as in natural as in artificial ant colonies, consists in pheromone trails that ants lay while searching for food. The structures that emerge from the accumulated pheromone trails serve as a sort of colony's external memory that can be used by any of its members. Even though a single ant has no knowledge of the global patterns, the designer of such a swarm-based system is a privileged observer of the emergence that comes as a result of the cooperative behavior.

The resulting mass behavior in swarms is hard to predict. Although the adaptability can be demonstrated on a variety of applications such as in image segmentation (Ramos and Almeida, 2000), a general theoretical framework on design and control of swarms does not exist. Artificial swarms use the bottom-up approach, meaning that the designer of such a system needs to set the rules for local interactions between the agents themselves and, if required, between the agents and the environment. The indirect communication via environment is referred to as *stigmergy*, and in case of ant colonies, it consists in pheromone-laying and pheromone-following. For each specific

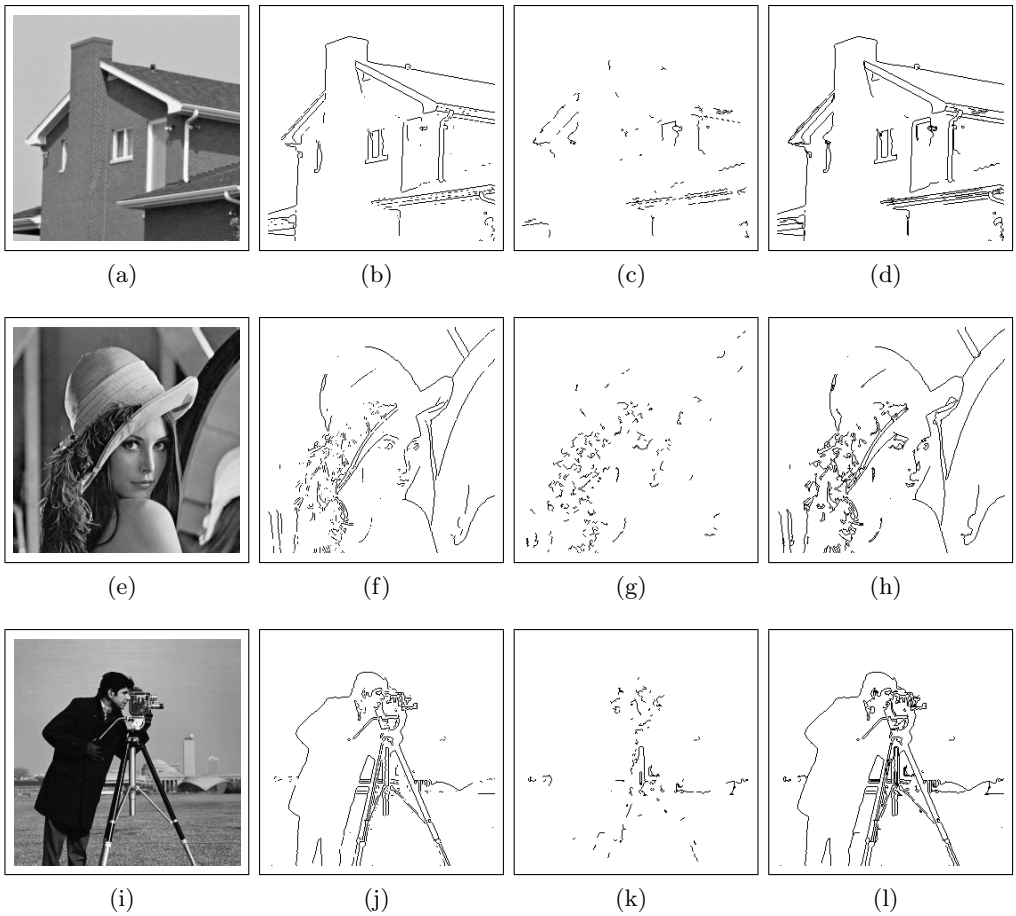


Figure 5.10: Qualitative results of the proposed method, 256 × 256-pixel images: (a) "House" original image; (b) "House": Sobel edge image; (c) "House": result of the proposed method; (d) "House" improved edge image; (e) "Lena" original image; (f) "Lena": Sobel edge image; (g) "Lena": result of the proposed method; (h) "Lena" improved edge image; (i) "Cameraman" original image; (j) "Cameraman": Sobel edge image; (k) "Cameraman": result of the proposed method; (l) "Cameraman" improved edge image.

application, the food that ants search for also needs to be defined.

This section presents a study on the adaptability of the algorithm proposed in Section 5.3 (Jevtić and Andina, 2010). Experiments with two different sets of grayscale images were performed. In the first experimental setup, a set of three different

grayscale images was used to test the adaptability of the proposed AS-based edge detector. The images were obtained by applying a Multiscale Adaptive Gain contrast enhancement to the 256×256 pixel "Cameraman" image (see Fig. 5.4). Every $N_i = 100$ iterations one image from the set was replaced by another. The response of the artificial ant colony to the change in the environment was a different distribution of pheromone trails. The number of 100 iterations per image was enough for the new pheromone structure to be established. The algorithm parameters used in the experiments were determined empirically: $\tau_0 = 0.01$, $\rho = 0.05$, $\alpha = 2$, $\beta = 1$, $T = 0.08$ and the tabu list length was set to be 10. The parameters could be optimized for a better edge detection, but it is of no importance for this study. It would not affect the adaptability of the algorithm since every image change would result in a change of the pheromone trail structure. The simulation results are shown in Fig. 5.11.

The results show that the Ant System-based edge detector was capable of detecting the changes that occurred as a result of replacing one image from the set with another. The experiments were repeated for a set of four widely used test grayscale images: "Cameraman", "Lena", "House", and "Peppers". The images were used as inputs to the algorithm in that order. Every $N_i = 100$ iterations one image was replaced by the next one from the set. Again, the change in the environment produced by the change of input image resulted in different pheromone patterns, which is shown in Fig. 5.12.

It can be observed that the new pheromone trails accumulated on the pixels belonging to the newly-emerged edges, while the pheromone trails where the edges were no longer present gradually disappeared. In order to accelerate the changes in pheromone distribution, the evaporation rate ρ was set to a lower value ($\rho = 0.05$). This caused disappearing of the "weakest" edges and introduced slightly poorer overall performance of the proposed edge detector. The experimental results show that the algorithm is able to adapt to a dynamically changing environment resulting in different pheromone trail patterns. Even though the images were used in the experiments, the study could be extended to any other type of digital habitat which can lead to a new set of applications for the adaptive artificial ant colonies.

One of the possible applications for the adaptive edge detector could be real-time image processing where online image preprocessing could be used to obtain better image segmentation. By applying various image enhancement techniques, such as contrast enhancement, certain features in the image could be amplified while others could be reduced or even removed. This would enable easier detection of the regions of interest in the image.

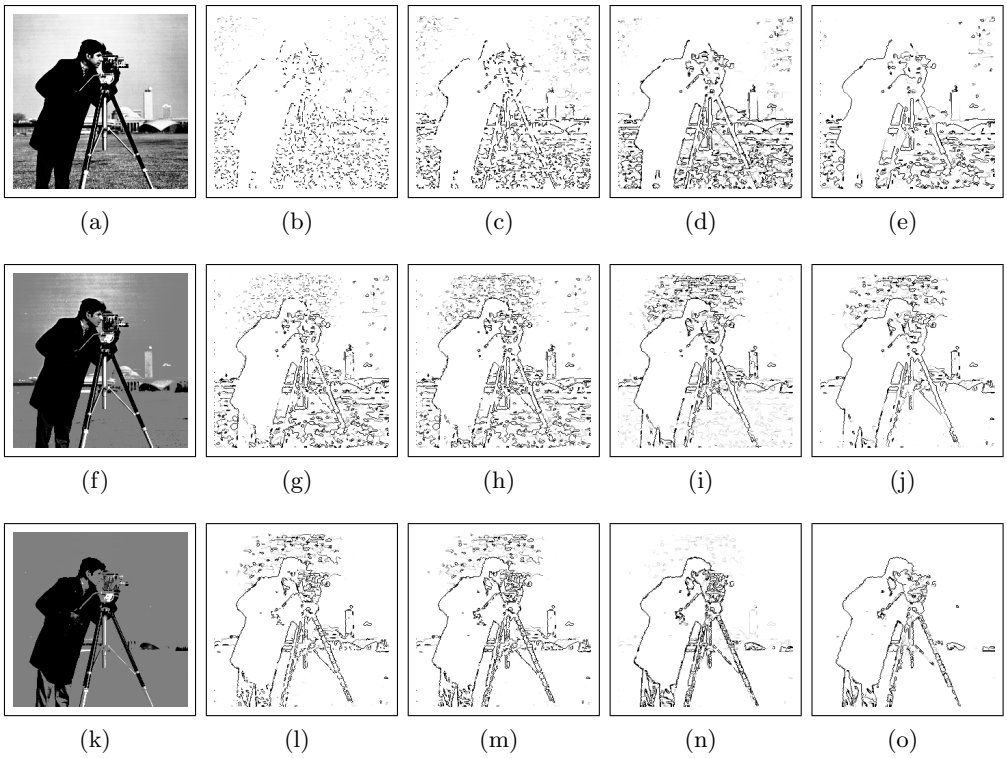


Figure 5.11: Adaptive edge detection on enhanced "Cameraman" images, 256×256 pixels: (a) enhanced image 1; (b) t=5 iterations; (c) t=10 iterations; (d) t=50 iterations; (e) t=100 iterations; (f) enhanced image 2; (g) t=105 iterations; (h) t=110 iterations; (i) t=150 iterations; (j) t=200 iterations; (k) enhanced image 3; (l) t=205 iterations; (m) t=210 iterations; (n) t=250 iterations; (o) t=300 iterations.

5.6 Summary

In this chapter, two methods for edge detection in grayscale images were proposed. The methods are based on the Ant System algorithm that models the foraging behavior of ant colonies. The model of the artificial ant colony is created with a bottom-up approach, using the rules of local interactions between the ants and the environment (digital image). This model is decentralized, self-organized, autonomous and adaptive to the changes in the environment. The adaptability is an important feature for the systems that are designed to be autonomous.

The learning ability, as in natural as in artificial ant colonies, consists in pheromone

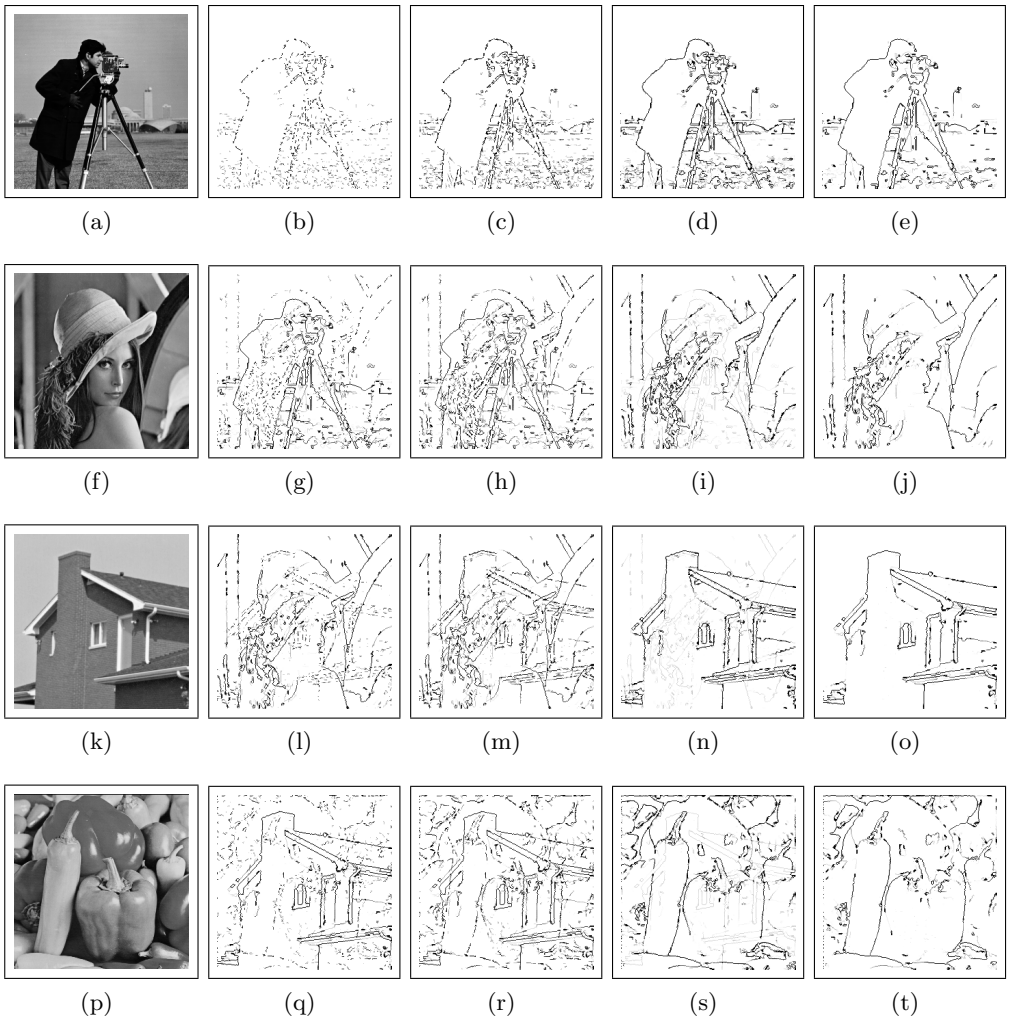


Figure 5.12: Adaptive edge detection on four test images, 256 × 256 pixels: (a) "Cameraman"; (b) t=5 iterations; (c) t=10 iterations; (d) t=50 iterations; (e) t=100 iterations; (f) "Lena"; (g) t=105 iterations; (h) t=110 iterations; (i) t=150 iterations; (j) t=200 iterations; (k) "House"; (l) t=205 iterations; (m) t=210 iterations; (n) t=250 iterations; (o) t=300 iterations; (p) "Peppers"; (q) t=305 iterations; (r) t=310 iterations; (s) t=350 iterations; (t) t=400 iterations.

trails that ants lay while searching for food. The structures that emerge from the accumulated pheromone trails serve as the colony's external memory that can be used

by any of its members. Even though a single ant has no knowledge of the global patterns, the designer of such a swarm-based system is a privileged observer of the emergence that comes as a result of the cooperative behavior. The edge detection in digital images makes an ideal application to visualize these patterns.

Chapter 6

Cluster Analysis for Image Segmentation

6.1 Introduction

Image segmentation is an important preprocessing step in applications of computer vision. The objective is to partition the image into homogeneous regions that share certain visual characteristics. Various image segmentation methods have been proposed which can be grouped in different classes such as clustering, edge detection, region growth, histogram-based, etc. (Gonzalez and Woods, 2008). No matter what method is applied, the objective remains the same, to change the representation of the original image into something easier to analyze.

The result of image segmentation are pixels grouped in clusters based on their similarities. There is therefore a natural tendency to apply data clustering algorithms to image segmentation. Clustering is a method of unsupervised learning because unlike classification, no prior labeling of data is available (Jain et al., 1999). The objective of the clustering process is to find groups of pixels that are similar in terms of a predefined characteristic, such as gray level intensity. A cluster of pixels is usually associated with a prototype, the most representative pixel, which is considered the center of the cluster. The similarity of a pixel with the prototype is evaluated according to their mutual distance and each pixel is assigned to the nearest prototype.

In some computer vision applications such as mammography for the cancer risk analysis, the less representative pixels are precisely the most interesting because they represent a variation with respect to healthy tissue. The pixels of high gray-level intensity could indicate the presence of microcalcifications which may be an early

sign of breast cancer. These pixels are difficult to detect because they can appear in groups of just a few pixels, for which are often merged with larger clusters. Various clustering methods proposed in literature differ in sensitivity to detect small regions of interest. A different approach, as proposed in this chapter, is to use pheromone accumulation inherent to ant colonies in nature to map the data space in order to make it easier to isolate these regions.

For the image segmentation the Ant System-based Clustering Algorithm (ASCA) is proposed, which extracts the clusters of pixels with a similar intensity level of gray. The ASCA is used to automatically determine the number of different clusters. Our algorithm, as its name suggests, is based on the Ant System algorithm (Dorigo et al., 1996) initially proposed to solve combinatorial optimization problems such as Traveling Salesman Problem. The Ant System models the pheromone-laying pheromone-following behavior of ants when they move in a discrete data space, which in our case is a set of digital images. For the experiments, two types of images were used in which the imperfections can be a result of light reflection over the image or the real imperfections that can be used as an aid in medical diagnosis. The images are represented in gray levels and, particularly, only one characteristic corresponding to the intensity level of each pixel is used.

The chapter is organized as follows. Section 6.2 provides a summary of the related work. In Section 6.3 the proposed ASCA algorithm is described. The image segmentation based on the proposed algorithm and the experimental results are presented in Section 6.4. Discussion on the obtained results and comparison with state-of-the-art clustering algorithms are also given in this section. Finally, in Section 6.5 the summary of this chapter is given.

6.2 Related Work

In this section, an overview of the state-of-the-art clustering algorithms is given. For image segmentation, clustering can be considered a preprocessing step that does not include the spatial information of the pixels. Therefore, the goal of the clustering process is to group the pixels based on their similarities in order to facilitate further knowledge extraction. Kotsiantis and Pintelas (Kotsiantis and Pintelas, 2004) define the following five categories of clustering algorithms: partitioning methods, hierarchical methods, density-based methods, grid-based methods and model-based methods. The swarm-based algorithms do not explicitly belong to any of the named categories (Handl and Meyer, 2007).

The Self-Organizing Maps (SOM) (Kohonen, 1990), an Artificial Neural Network (ANN) with unsupervised learning, is a widely used clustering algorithm. SOM is useful for data classification because of its visualization property. For example, the SOM was used for pattern recognition in satellite images (Richardson et al., 2003), or segmentation of color images (Jiang and Zhou, 2004), but also many others.

A cluster of pixels is usually associated with the prototype as the most representative pixel also considered the cluster center. The partitioning methods use this centric property to divide N-dimensional data space, where the partitions are either strict, fuzzy or possibilistic. The most well-known strict partitioning algorithm is the k-Means (MacQueen, 1967) which divides a data set into k subsets such that all points in a given subset are closest to the same center. The drawback of the k-Means algorithm is that it did not provide any information on how close to each prototype a data point was. Therefore, Bezdek (Bezdek, 1981) proposed the Fuzzy c-Means algorithm (FCM) that calculates a membership degree for each data point in relation to different clusters, where the sum of the membership degrees of a point must be equal to one. The FCM was insensitive to noise since several equidistant data with the same membership values are not equally representative of the clusters. The solution was offered with the Possibilistic c-Means algorithm (PCM) algorithm (Krishnapuram and Keller, 1993) that identifies the similarity of data with a given number of prototypes using typicality values ranging from (0, 1). The main drawback of this algorithm is that the clustering results are dependent on the manually-set typicality threshold. Pal et al. proposed to use both membership degrees and typicality values and implemented it in the Fuzzy Possibilistic c-Means (FPCM) algorithm (Pal et al., 1997). Another improvement was proposed by the same author as the Possibilistic Fuzzy c-Means (PFCM) algorithm (Pal et al., 2005) where he introduced control parameters which define the relative importance of the membership degrees and the typicality values.

The clustering performance of the partitioning-based algorithms is greatly dependent on the initial guess of cluster centers and it is time consuming. Various methods were proposed to address these issues. One simple implementation of the Lloyd's k-Means algorithm to color quantization, data compression and image segmentation was proposed in (Kanungo et al., 2002). The algorithm stores the multidimensional data points in a kd-tree that is computed only once, which results in faster computation. Laia and Liaw (Lai and Liaw, 2008) proposed a modified k-Means algorithm to speed up the clustering process for larger data sets with higher dimension. Many other partitioning-based methods were proposed to achieve a faster execution (Wang

and Rau, 2001; Junwei and Yongxuan, 2007), but also for a more robust and less noise-sensitive clustering (Liew et al., 2000; Chang and Yeh, 2005; Li et al., 2007; Awad et al., 2009).

The sensitivity in detection of the atypical data remains an issue for the state-of-the-art clustering algorithms. Ojeda *et al.* (Ojeda-Magaña et al., 2009) propose an image sub-segmentation method based on the PFCM algorithm in order to detect small homogeneous regions in mammograms. The authors applied a typicality value threshold to delimit a sub-group containing atypical pixels within the initially detected clusters. The threshold value was set manually which is the main drawback of the proposed PFCM-based method. Another drawback, common to partitioning-based methods, is that the number of clusters has to be *a priori* known.

Various methods proposed in literature suggested the use of pixel spatial information to improve the image segmentation results. Although it is beyond the scope of the here-proposed work, some of these methods are mentioned. Chuang *et al.* (Chuang et al., 2006) proposed an image segmentation method based on the FCM algorithm. The authors incorporated the spatial information into the membership function of each pixel with regards to the neighboring pixels in order to detect more homogeneous regions and make the algorithm less sensitive to noise. Cai *et al.* (Cai et al., 2007) proposed the Fast Generalized Fuzzy c-Means (FGFCM) algorithm for more robust and faster image segmentation. Their method is based on improvement of other FCM-based algorithms to include the local spatial and grey information in the membership function.

6.2.1 Ant Clustering Algorithms

The group of data clustering algorithms inspired by the swarming behavior of natural ant colonies is usually referred to as *ant clustering algorithms*. Although these algorithms are based on the idea of the cooperative behavior in nature, in many cases their biological plausibility is sacrificed in order to obtain better clustering results. Handl and Meyer (2007) roughly classify the ant clustering algorithms in two main groups. The first group mimics the corpse gathering and brood sorting in natural ant colonies, where the clustering solution emerges as a result of interactions of the ants with the environment. The second group of algorithms uses general-purpose ant-based optimization methods in order to obtain optimal clustering of the given data set. There are also algorithms that do not explicitly belong to any of these groups, but fall somewhere in between.

A basic algorithm which mimics the clustering behavior of an ant colony was

proposed by Lumer and Faieta (1994) and it was based on the model by Deneubourg et al. (1991) who studied the way the ant colonies group the corpse inside their nest. The authors proposed that the N-dimensional data set be randomly distributed on a two-dimensional grid. The virtual ants move around the grid randomly picking and dropping the data items. The decision of either picking or dropping the item at the specified position on the grid depends on the similarity of the other neighboring items.

Several authors have built their algorithms on the model proposed by Deneubourg et al. Some introduced modifications to Lumer and Faieta algorithm in order to improve its performance and runtime (Handl and Meyer, 2002; Vizine et al., 2005). Another version of the algorithm called ATTA-C (Handl et al., 2006), although improved, failed to perform effectively on the designated application, as it was proven by the authors.

Interestingly, Martin et al. (2002) observed that there was no collective effect in the clustering model proposed by Deneubourg et al., which implies that the algorithms derived from this model do not demonstrate the collective intelligence. Tan et al. (2006) removed the collective element from the mentioned clustering model to prove that the speed and the quality of solution did not depend on the number of ants used in experiments.

The ant clustering algorithms can be used to automatically identify the number of clusters inherent to a data collection. This feature is of utmost importance when the global knowledge of the data set does not exist. In (Handl et al., 2006), the authors showed that the ATTA-C algorithm outperforms Gap statistic algorithm (Tibshirani et al., 2000) in terms of extraction of the number of clusters when the distance between the cluster centers decreased.

Another type of ant clustering algorithms is based on the general-purpose ACO (Dorigo and Blum, 2005). These algorithms require post-processing to extract the explicit clusters from the pheromone matrix. One of the early algorithms that applied ACO to find a cost-minimizing path between the nodes defined by data points were proposed by Tsai et al. (2002) and Chu et al. (2004). The ants were used to connect the nodes on their path with the pheromone trails. A threshold was used to remove the week node connections and the remaining ones formed the clusters.

Some authors proposed combining the ACO algorithms with other clustering methods such as the standard K-means (Saatchi and Hung, 2005) or the Fuzzy C-Means (Runkler, 2005) to make them less dependent on initial conditions such as the number and the position of the cluster centers. Yu et al. (2009) proposed an image

segmentation method using an ant clustering algorithm that applies fuzzy rules to determine the membership function of a pixel. The drawback of this method is that it requires the number of clusters to be *a priori* known and its high computational cost.

The ASCA algorithm proposed in this chapter belongs to the group of ant clustering algorithms that use general-purpose ant-based optimization methods, in this case to find the optimal routes from nodes to their closest cluster centers. A detailed description of the algorithm is given in the following section.

6.3 Ant System-based Clustering Algorithm (ASCA)

In this section, the Ant System-based Clustering Algorithm (ASCA) is described (Jevtić et al., 2011b). As its name suggests, it is based on the AS algorithm, which was inspired by the foraging behavior of ant colonies in nature. When ants find a food source, they leave pheromone trails that attract other ants to follow their path. Pheromone trails evaporate over time, so a path that leads to a closer food source accumulates more pheromone as it is crossed by ants more frequently. The Ant System algorithm exploits this cooperative behavior of ant colonies that features indirect communication through the environment. Unlike their biological counterparts, the artificial ants move through a discrete space defined with nodes and they have memory of the taken path.

Pheromone trails in the ASCA algorithm are accumulated in nodes in order to represent the density of the surrounding data. This differs from what was proposed in basic Ant System algorithm where pheromone trails marked the edges that connected the nodes in order to represent the favorite path. The process of pheromone accumulation is iterative and creates a pheromone map of the data set to be clustered. Higher data-density areas accumulate more pheromone and they represent cluster centers. This is used to extract the number of clusters. Gradient of the pheromone trail is used to assign every node to a cluster by applying local Hill climbing search (Russell and Norvig, 2003). The ASCA algorithm consists of three consecutive parts, namely: a) pheromone accumulation, b) local pheromone summing, and c) data labeling.

6.3.1 Pheromone Accumulation

In the pheromone accumulation stage, the artificial ants move in N-dimensional data space looking for the high data-density regions. The algorithm starts with an initialization step which is followed by the iterative construction of new solutions and

pheromone update. It involves the following steps:

1. Initialization: All nodes are initialized with an equal small amount of pheromone, τ_0. The population of M ants is created and placed on randomly chosen nodes.

2. Node transition rule: Ant chooses the next node to move to by applying the roulette rule. That is, every node has associated probability with which it is chosen from a set of available nodes. The probability of displacing kth ant from node i to node j depends on the Euclidean distance between the nodes and the amount of pheromone trail accumulated in node j, and it is given by:

$$p_{ij}^k = \begin{cases} \frac{(\tau_j)^\alpha (\eta_{ij})^\beta}{\sum_{h \notin tabu_k, (\tau_h)^\alpha (\eta_{ih})^\beta}}, & \text{if } j \notin tabu_k \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

   where τ_j and η_{ij} are the intensity of the pheromone trail on the node j and the visibility of the node j with respect to the node i, respectively. The visibility is given as the reciprocal value of the Eucludean distance, d_{ij}, between the nodes i and j:

$$\eta_{ij} = \frac{1}{d_{ij}}. \tag{6.2}$$

   Ant is not allowed to displace to the nodes it has already visited. $Tabu_k$ list contains the nodes visited by the kth ant. The control parameters α and β allow us to bias the decision-making mechanism towards the exploitation of the generated knowledge about the environment or exploration of new solutions, respectively. ($\alpha, \beta > 0$; $\alpha, \beta \in \Re$.) It can be noticed that the accumulated pheromone trails serve as a colony's external memory where the extracted knowledge about the environment is stored.

3. Pheromone update rule: Once all the ants carry out the transition to other nodes, the pheromone update is applied to each node as follows:

$$\tau_{j,new} = (1 - \rho)\tau_{j,old} + \sum_{k=1}^{M} \Delta\tau_j^k \tag{6.3}$$

   where ρ is the pheromone evaporation rate ($0 < \rho < 1$; $\rho \in \Re$), and $\Delta\tau_j^k$ is the amount of pheromone laid on the node j by the kth ant, and is given by:

$$\Delta\tau_j^k = \begin{cases} \eta_{ij}, & \text{if node } j \text{ has been visited by } k\text{th ant} \\ \\ 0, & \text{otherwise.} \end{cases} \tag{6.4}$$

where η_{ij} is the visibility of the node j from the node i from which the kth ant was displaced as defined in (6.2).

4. Stopping criterion: The steps 2 and 3 are repeated in a loop and the algorithm stops executing when the maximum number of iterations is reached.

The output is a pheromone matrix where the distribution of pheromone is scarce, but the highest concentrations are found around the dense regions of nodes in data space. The ants make a probabilistic choice of path, therefore the neighboring nodes may have significantly different amounts of pheromone deposits (see Fig. 6.1(b)). For this reason, a local pheromone summing is applied.

6.3.2 Local Pheromone Summing

The pheromone trails are locally summed to obtain a smooth pheromone surface to which a local gradient-based search will be applied. Lets define the neighborhood resolution as an N-dimensional sphere whose diameter is calculated as a portion of the Euclidean data space, and is given by:

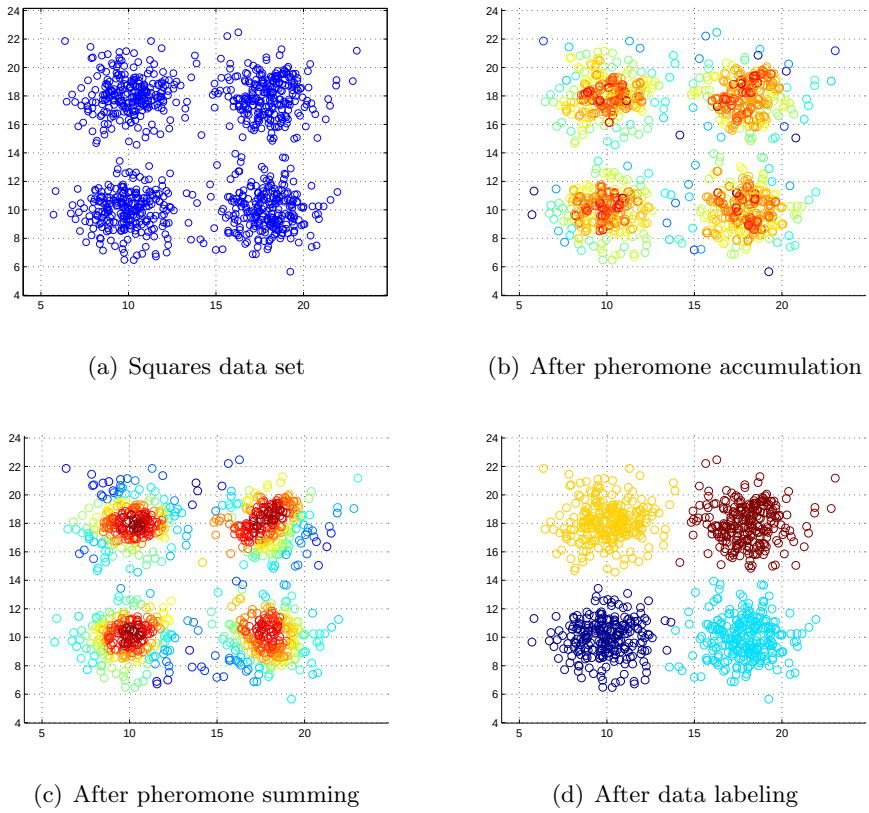
$$r = \frac{1}{\gamma}\sqrt{\sum_{n=1}^{N}(x_{n,max} - x_{n,min})^2} \tag{6.5}$$

where γ is the resolution ratio, and $(x_{n,min}, x_{n,max})$ is the data space range for the nth dimension. In order to update the pheromone trails, to the pheromone value in node i, τ_{i0}, are added the pheromone values from its neighboring nodes, as follows:

$$\tau_{is} = \tau_{i0} + \sum_{n=1}^{N_{neigh}} \tau_n \tag{6.6}$$

where N_{neigh} is the number of neighboring nodes contained within the sphere defined in (6.5).

The output of the local pheromone summing process is a smooth (N+1)-dimensional surface with pheromone value peaks around the cluster centers (see Fig. 6.1(c)).



(a) Squares data set

(b) After pheromone accumulation

(c) After pheromone summing

(d) After data labeling

Figure 6.1: Clustering result after applying the proposed ASCA algorithm to a Squares data set, which consists of four groups of 250 data patterns with normal Gaussian distribution.

6.3.3 Data Labeling

In the final step of the ASCA algorithm a discrete Hill-climbing search is applied, a gradient-based search, to find the local maxima on the pheromone surface. The search is performed from each node with the step size defined in (6.5). All the nodes from which the search results with the same local maximum are grouped in one cluster (see Fig. 6.1(d)). The number of clusters is therefore equal to the number of local maxima. It is important to emphasize the property of the ASCA algorithm to extract the number of clusters from the data set, unlike the algorithms that it is compared with that require this number to be known in advance.

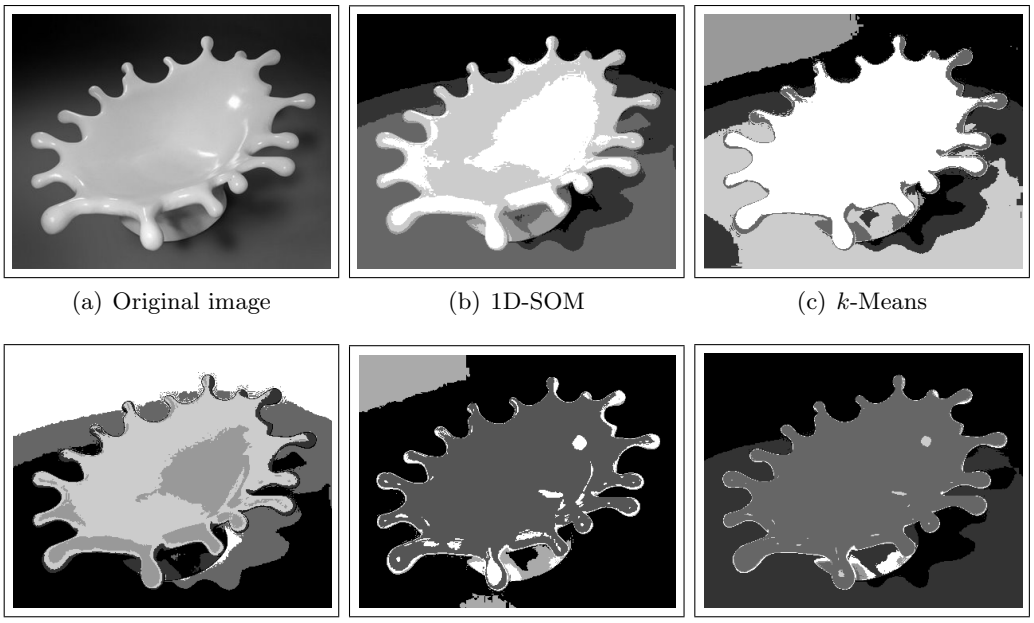
6.4 Experimental Results and Discussion

In this section, the proposed ASCA algorithm to the image segmentation is applied. The advantage is taken from the distributed organization of the artificial ant colony and the pheromone patterns that emerge as a result of local interactions in a discrete space of digital images. Specifically, pheromone mapping of a set of unlabeled image pixels characterized by the gray intensity level is obtained in order to cluster them. The objective is to detect the atypical pixels and for that reason two kinds of images were used, the "Splash" image, see Fig. 6.2(a), and a mammogram, see Fig. 6.3(a) and Fig. 6.4(a).

The selected values for the parameters of the algorithm, for all the images, are: $\tau_0 = 100$, $\alpha = 1$, $\beta = 5$, $\rho = 0.05$, $\gamma = 30$. The experiments were performed in 3 cycles with 30 iterations. In each cycle, a population of 1000 ants were displaced on randomly chosen nodes. The experiments were performed using MATLAB (software MATLAB, version R2009b), on a computer with Pentium IV processor at 3.4 GHz, with 2 GB of RAM. For the k-Means and FCM algorithms, the functions already implemented in MATLAB were used. The settings for 1D-SOM were implemented as proposed in (Barrón-Adame et al., 2007) and the results of PFCM clustering were obtained from (Ojeda-Magaña et al., 2009). The results of simulations are shown in Fig. 6.2, for the "Splash" image, and Fig. 6.3 and Fig. 6.4 for a region of interest (ROI) mammogram.

For the "Splash" image shown in Fig. 6.2 the objective was to detect the pixels of high gray-level intensity that are a result of light reflection. The ASCA algorithm extracted six clusters which was enough to detect the atypical pixels. A comparison was made with 1D-SOM, k-Means, FCM and PFCM algorithms using the same number of clusters (six). The ASCA outperformed 1D-SOM, k-Means and FCM which were not able to extract the regions of interest. In case of the PFCM, image sub-segmention was applied after the initial segmentation in two pixel groups. Because of the limitation of this approach to divide the data space in 2^n partitions, the segmentation gave four clusters. Although the PFCM managed to detect the light reflection pixels, some other features like the shadow of the splash were not extracted. As a second part of the experiment, though not presented here, for each algorithm the number of clusters was incremented until the image segmentation allowed the separation of light reflection pixels. The following results were obtained: 1D-SOM, 17 clusters; k-Means, 9 clusters; and FCM, 7 clusters.

In mammography, the presence of microcalcifications could be an early sign of

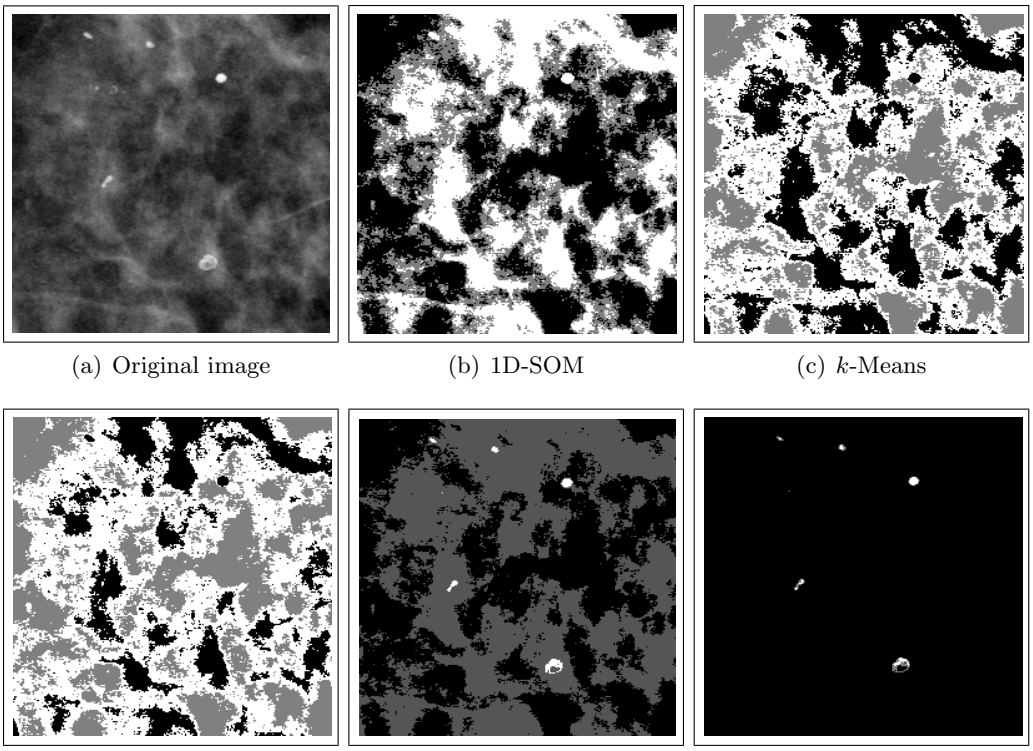


(a) Original image (b) 1D-SOM (c) k-Means

(d) FCM (e) PFCM (f) ASCA

Figure 6.2: Comparison of the segmentation results for the "Splash" image, 320 × 400 pixels. The ASCA extracted six clusters; for the purpose of comparison, other algorithms were set to partition the data set in six clusters as well, only in case of PFCM (Ojeda-Magaña et al., 2009) because of the limitation of the algorithm to have 2^n partitions the results are shown for four clusters.

breast cancer. They appear as small groups of pixels of high gray-level intensity and they usually occupy a very small range of values, hence they are hard to detect. The ROI mammograms on which the experiments were performed and the results of image segmentation using 1D-SOM, k-Means, FCM, PFCM and the proposed ASCA algorithm are shown in Fig. 6.3 and Fig. 6.4. The ASCA algorithm extracted three clusters from the first (Fig. 6.3(f)) and five clusters from the second image (Fig. 6.4(f)). For comparison, 1D-SOM, k-Means and FCM were set for three and five clusters in the first and second image, respectively. The image segmentation using these algorithms gave poor results as the microcalcification pixels could not be isolated. The PFCM managed to detect the microcalcifications in the first ROI mammogram (Fig. 6.3(e)) but with a higher number of clusters (four) than the ASCA. In the second ROI mammogram PFCM isolated the region of interest but it also detected a larger



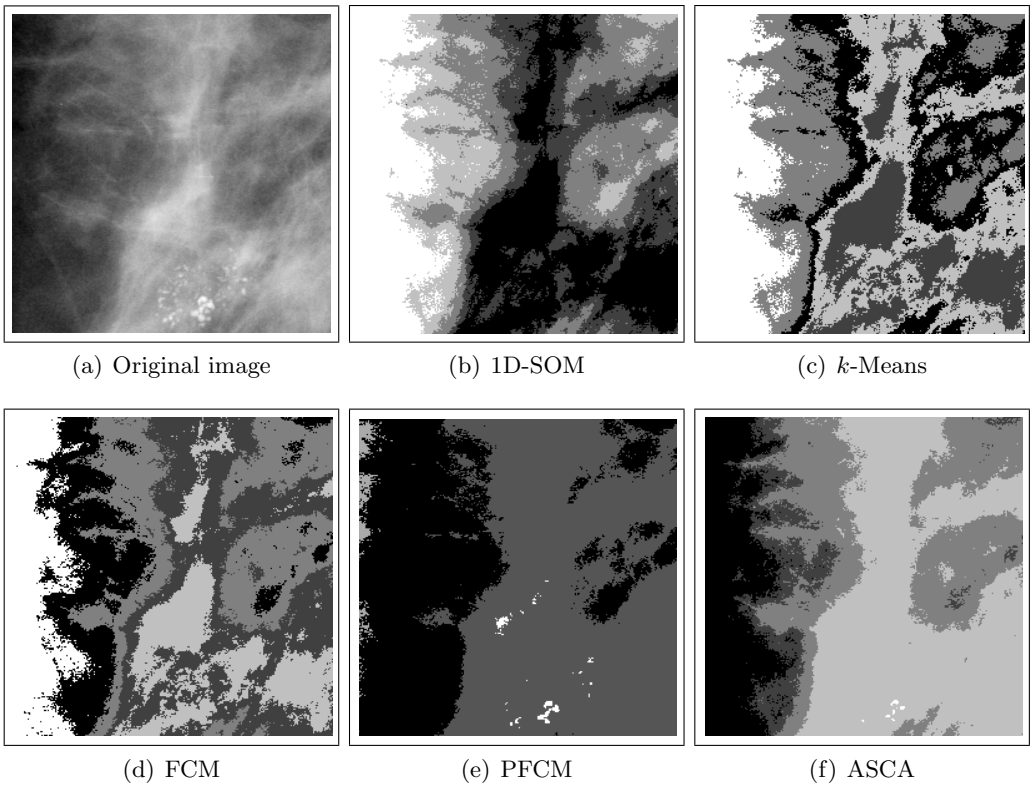
(a) Original image (b) 1D-SOM (c) k-Means

(d) FCM (e) PFCM (f) ASCA

Figure 6.3: Comparison of the segmentation results for a ROI mammogram, 256×256 pixels. The ASCA extracted three clusters; for the purpose of comparison, other algorithms were set to partition the data set in three clusters as well, only in case of PFCM (Ojeda-Magaña et al., 2009) because of the limitation of the algorithm to have 2^n partitions the results are shown for four clusters.

group of pixels in the central part of the image that belonged to a healthy tissue (see Fig. 6.4(e)). By increasing the number of clusters, in case of 1D-SOM, k-Means and FCM, it was not possible to obtain better segmentation results. The output images became over-segmented which prevented the extraction of the regions of interest.

For the ASCA algorithm, image pixels clustering comes as a result of the adaptive behavior of artificial ant colony which finds paths from the peripheral regions of a cluster to its center accumulating there higher concentration of pheromone. The indirect interaction between the ants via environment gives different results from what



(a) Original image (b) 1D-SOM (c) k-Means

(d) FCM (e) PFCM (f) ASCA

Figure 6.4: Comparison of the segmentation results for a ROI mammogram, 256 × 256 pixels. The ASCA extracted five clusters; for the purpose of comparison, other algorithms were set to partition the data set in five clusters as well, only in case of PFCM (Ojeda-Magaña et al., 2009) because of the limitation of the algorithm to have 2^n partitions the results are shown for four clusters.

would be obtained by simple data-density calculation. Not all high-density regions accumulate equal amounts of pheromone even though their local neighborhood may be similar (e.g. same distance from the nearest neighbor). The clusters' centers are crossed by more ants, from various directions; therefore, they will accumulate more pheromone than the peripheral regions. In order to apply the proposed algorithm to real-time applications, the algorithm's code should be optimized for a faster execution, preferably not in Matlab but in another programming environment.

6.5 Summary

The important feature of the proposed ASCA algorithm is automatic extraction of number of clusters. This is useful for applications where the groups of patterns within a data set are not well-defined and need to be detected. The performance of the ASCA algorithm was evaluated through experiments on two types of images. Our algorithm outperformed 1D-SOM, k-Means, FCM and PFCM algorithms in detection of small, atypical regions of the image, either in the overall quality of segmented images or the total number of clusters used.

The resolution ratio γ is used to set the sensitivity of the algorithm to the density of the data. Future work will include the optimization of the algorithm's parameters, namely the resolution ratio γ and control parameters α and β, in order to improve the adaptability of the algorithm to different data sets. For the case of image processing, more features such as spatial information or texture will be used to obtain better segmentation results. Also, testing the algorithm on the color images may open a wide range of new applications.

Part III

Multi-Agent System Modeling

Chapter 7

Distributed Task Allocation in Swarm of Robots

7.1 Introduction

In applications that are too risky or too demanding for humans, or where a fast response is crucial, multirobot systems can play an important role thanks to their capability to cover the area. Possible applications are planetary exploration, urban search and rescue, monitoring, surveillance, cleaning, maintenance, among others. In order to successfully perform the tasks, robots require a high degree of autonomy and a good level of cooperation. The set of robots should behave like a team and not merely as a set of entities.

In scenarios that require area coverage, dozens, hundreds or even thousands of robots can be used. Such a large group of robots, if organized in a centralized manner, could experience information overflow that can lead to the overall system failure. For this reason, the communication between the robots can be realized through local interactions, either directly with one another or indirectly via environment. The large group of robots acting in such a manner is referred to as swarm (Beni and Wang, 1989).

As a result of the growing interest in coordination of multirobot systems, multirobot task allocation (MRTA) has become an important research topic. The goal is to assign the tasks to the robots in a way that, through cooperation, the global objective is achieved more efficiently. In the scenario proposed in this work, tasks are represented by targets defined by their qualities and their location in the robot arena. A simulator was developed that implements a multi-foraging scenario, and the

experimental setup addresses the issue on how to, in a distributed way, assign the robots to the found targets with a expected distribution determined from the targets qualities.

Scalability in its most general form is defined as how well a solution to some problem will work when the size of the problem increases. In the context of mobile multirobot systems, scalability refers to the overall system's performance if the number of robots increases in relation to the number of tasks at hand (Rana and Stout, 2000). The resulting effect on the system's performance can be determined in terms of metrics associated with a particular platform or an operating environment, which in our work refers to dispatching a robot to a remote site marked as a target.

For a swarm of robots engaged in a multi-foraging scenario the Distributed Bees Algorithm (DBA) was proposed (Jevtić et al., 2010b), which was inspired by the foraging behavior of colonies of bees in nature. When they find a food source, the scout bees return to the hive and perform a famous "waggle dance" in order to recruit other bees. The information about the richness and location of the source is passed using direct communication. Some models of the cooperative behavior of bee colonies with centralized communication or no communication have already been proposed (Pham et al., 2006; Bailis et al., 2010). In order to avoid the centralized concept of the beehive dance floor, our robots were designed to use broadcast communication to inform other robots in the range about the estimated location and the quality of the found target.

The objective of the proposed algorithm is to assign the robots in a swarm to the found targets in such a way that the final distribution is proportional to the targets' qualities. The targets with associated qualities represent a distributed "food" that requires a usually non-uniform distribution of robots in the area. The algorithm has been previously validated through experiments with real robots (Jevtić et al., 2010b). This chapter presents the analysis of the proposed algorithm's scalability in a simulated environment. The simulations were performed for various sets of parameters, including number of robots, number of targets and targets' quality values. Control parameters inherent to the DBA were tuned to test how they affect the final robot distribution.

The rest of the chapter is organized as follows. Section 7.2 provides a summary of the related work. In Section 7.3, a problem statement is given and a general mechanism for distributed task allocation in multi-foraging domain is described. In this section, a mathematical model of the proposed DBA algorithm is presented. In Section 7.4 the experimental setup with real robots and simulator are described. The

results from experiments are presented and discussed. The simulation environment is defined and the experimental evaluation for the multi-robot system's scalability is proposed. The statistical analysis of the simulation results is presented, and a discussion of the approach and the results is given. Finally, the Section 7.6 gives a summary of contributions.

7.2 Related Work

Multirobot systems offer the possibility of enhanced task performance, increased task reliability and decreased cost over more traditional single-robot systems. However, multirobot systems must be designed having these issues in mind. Research field of multirobot systems is not new and various architectures that differ in size and complexity have been proposed. Dudek et al. (2002) provided a taxonomy that categorizes the existing multirobot systems along various axes, including size (number of robots), team organization (e.g., centralized vs. distributed), communication topology (e.g., broadcast vs. unicast), and team composition (e.g., homogeneous vs. heterogeneous).

Rather than characterize architectures, Gerkey and Matarić (2004) categorized instead the underlying coordination problems with a focus on multirobot task allocation (MRTA). They distinguish: single-task (ST) and multi-task (MT) robots, single-robot (SR) and multirobot (MR) tasks, and instantaneous (IA) and time-extended (TA) assignment. The authors showed that many MRTA problems can be viewed as instances of well-studied optimization problems in order to analyze the existing approaches, but also to use the same theory in the synthesis of new approaches. In order to estimate a robot's performance, they defined utility that depends on two factors, namely expected quality of task execution and expected resource cost. Given a robot R and a task T one can define Q_{RT} and C_{RT} as the quality and cost, respectively, expected to result from the execution of T by R. The resulting nonnegative utility measure is:

$$U_{RT} = \begin{cases} Q_{RT} - C_{RT} & \text{if } R \text{ is capable of executing} \\ & T \text{and } Q_{RT} > C_{RT}, \\ 0 & \text{otherwise.} \end{cases} \tag{7.1}$$

This however is not a strict definition of utility which is a flexible measure of performance and can entail arbitrary computation. The only constraint on utility estimators is that they must each produce a single scalar value that can be compared for the purpose of assigning robots for tasks. The problem addressed in this chapter is categorized as a "single-task robots, multirobot tasks, instantaneous assignment

(ST-MR-IA)", which Gerkey and Matarić proposed to be solved as a set partitioning problem. However, this requires the combined utilities of all the robots to be known in advance, which is not the case. The proposed method is described in detail in Section 7.3.

What follows is a survey of various multirobot system architectures that have been proposed for solving different problems. We tend to use the above mentioned taxonomies to categorize them.

One of the common approaches for solving the ST-SR and ST-MR problems is a market-based approach which uses auctioning mechanism for task allocation. Matarić et al. (2003) proposed four different strategies for dynamical task allocation in two different emergency-handling scenarios. The robots bid for tasks and decisions are made by auctioning. Authors concluded that there is no overall best strategy and that the success of a strategy is task-related. Michael et al. (2008) proposed a market-based approach for robots formation control. They associate multiple tasks with predefined spatial locations that define a formation.

A thorough overview of market-based approaches for MRTA is given by Dias et al. (2006). A common drawback of these approaches is the underlying auctioning mechanism which requires all the bids from the robots to be gathered at one auctioning point. Sometimes, when resources permit, markets can even behave in a centralized fashion over larger portions of the robot team to improve solution quality. The authors gave a summary of communication costs for various market-based approaches. The main advantage of the method we propose is that, although it imposes certain communication cost for sending the information of the found targets, the robots make decisions autonomously and in a distributed manner. This is not the case with market-based approaches that feature a partial distribution, where robots are divided into sub-teams that take decisions in a centralized manner. For this reason, scalability in market-based approaches is often limited by the computation and communication needs that arise from increasing auction frequency, bid complexity, and planning demands.

Environment exploration and mapping are common applications for multirobot systems. Franchi et al. (2009) proposed a Sensor-based Random Graph (SRG) method for cooperative robot exploration. They addressed the issue of system's performance with respect to exploration time and traveled distance. The authors showed that by adding more robots the system could scale-up, but its performance was highly dependent on the initial team deployment, giving better results when the robots started grouped in a cluster than if scattered in the environment. The robots used broadcast

communication with a limited range but the concept of decentralized cooperation was in question since the robots were programmed to gather in sub-teams that had to synchronize for local path-planning and collision-avoidance. This required intensive interchange of information including robot's ID and displacement plan.

Another approach proposed by Burgard et al. (2005) treats the unknown environment exploration as a ST-SR problem, where individual robots select a new target location based on its distance and utility. The map was divided in cells whose size was determined by the robot's visual range. The utility of each target location, i.e. cell, would decrease if more of its neighboring cells were assigned to other robots. To determine appropriate target locations for all the robots, the authors proposed an iterative algorithm. The drawback of this algorithm is its complexity and high computational cost. Although the experimental results show the advantages of collaboration, the proposed centralized approach cannot be applied if not all robots can communicate with each other.

Decentralized coordination of robots has various advantages over more traditional centralized approaches. It can be applied to reduce the communication burden on multirobot system (Ray et al., 2009), especially for large teams of robots. In some applications communication can be difficult to implement or no communication exists at all. Joordens and Jamshidi (2010) proposed a decentralized coordination for a swarm of underwater robots which is based on consensus control. Another decentralized strategy for dynamical allocation of tasks that requires no communication among robots was proposed by Berman et al. (2009). But often, as in case of multirobot area coverage (Schwager et al., 2009), the decentralized coordination and distributed decision-making is applied having one goal in mind, that the global objective is achieved more efficiently.

7.2.1 Bio-inspired Coordination of Multirobot Systems

Robot swarms are multirobot systems that typically consist of a large population of simple robots interacting locally with one another and with their environment (Bonabeau et al., 1999). These systems draw inspiration from animal swarms in nature but their design is not constrained by biological plausibility. Their main feature is decentralized coordination which results in a desired behavior that emerges from the rules of local interactions.

The self-organizing properties of animal swarms such as insects have been studied for better understanding of the underlying concept of decentralized decision-making in nature (Camazine et al., 2001) but it also gave new approach in applications to multi-

agent system engineering and robotics. Bio-inspired approaches have been proposed for multirobot division of labor in applications such as exploration and path formation (Groß et al., 2008), multi-site deployment (Berman et al., 2007), or cooperative transport and prey retrieval (Labella et al., 2006; Campo and Dorigo, 2007).

The bottom-up design topology inherent to bio-inspired multirobot systems provides them with one or more of the following features, such as being autonomous, scalable, robust and adaptive to changes in their environment. On the other hand, the collective behavior has emergent properties that give them the ability to produce unpredictable patterns. One way of dealing with the unpredictability issue is statistical analysis through experiments, as proposed in this chapter.

7.2.2 Scalability

Task allocation scenarios include a set of tasks that may have different priorities and require one or more robots to be assigned to their execution. A very important property of multirobot systems is the ability to scale-up with respect to the number of robots or the number of tasks at hand. However, scalability of multirobot systems and multi-agent systems in general has been analyzed from various perspectives including the total number of agents involved, the size of the communication data, the number of rules the agents operate with, or the agents' diversity (Rana and Stout, 2000).

In order to evaluate the scalability of a given multirobot system one needs to identify a performance metrics. Various MRTA methods exist but, to the best of our knowledge, a comprehensive analysis tool for the scalability of such methods has not been given. Some mathematical models that have been proposed could serve as guidelines in multirobot system design, but different scenarios to which these systems are applied usually do not permit us to maintain within the proposed framework.

Lerman et al. (2006) proposed a mathematical model for MRTA in dynamical environments. The authors assumed that robots were able to observe tasks in order to discriminate their types, but also to discriminate the tasks that other robots were assigned to. Robots had limited sensing capabilities and could not directly communicate. The lack of communication made the system more robust to failures, but also more susceptible to noise from the sensors, and requires more time for exploration of available tasks.

Top-down design methodologies apply the classical control theory for performance estimation of distributed agent-based systems. While it is possible to establish bounds on the system behavior and provide performance guarantees, they heavily rely on the available bandwidth for robot communication and they are more sensitive to noise.

The need for resources becomes even a bigger issue as the number of robots increase. There is therefore a very natural tendency to apply bottom-up methodologies which produce autonomous, scalable and adaptable systems requiring minimal communication (Crespi et al., 2008).

Broadcast communication provides quick propagation of tasks' information within the multirobot system but extensive use of communication channel can affect the system's scalability. Previously described market-based approaches suffer from a large requirement in terms of communication bandwidth as they use broadcast messages to auction for the tasks. Farinelli et al. (2006) proposed a mechanism based on token passing for cooperative object retrieval, which scales up for reliable sending of broadcast messages. The authors made a comparison of their method with market-based approaches and the ones based on iterative broadcast communication. Their results show that the ability of the system to adjust to the available communication bandwidth provides guarantees for better performance.

7.3 Distributed Task Allocation

7.3.1 Problem Definition

Based on the described taxonomy, our multirobot system can be categorized as homogeneous and distributed, using broadcast communication. We address a problem of single-task robots, multirobot tasks and instantaneous assignment (ST-MR-IA). The task allocation scenario studied here considers the environment that contains a number of tasks that could be of same or different importance and robots that are equally capable of performing each task but can only be assigned to one at any given time. More specifically, the tasks are targets with their associated qualities. The quality of a target is an application-specific scalar value that may represent target's priority or complexity, where a higher value requires more robots to be allocated. For example, it could represent the richness of the mineral or water source on a planet needed to be harnessed, the amount of garbage to be collected in a public space, or the number of injured people in a need for assistance in urban search and rescue scenario. In this work, it is not consider how these values were obtained.

The proposed scenario is presented under the following assumptions:

1. All the targets are made available to all the robots. This is done by setting the broadcast communication range of the robots to cover the entire arena.

2. Robots take decisions once all the targets in the arena are found, unless they

were the ones that found a target in which case they are automatically allocated to that target. The total number of targets is preset in robots' internal memory and it depends on the experimental setup.

3. Reallocation to another task is not allowed.

These assumptions are taken for simplicity; otherwise, it would be difficult to analyze the performance of the system due to the unpredictability of the robots' distribution prior to task allocation.

Consider a population of N robots to be allocated among M targets. Let $Q \in \{q_1, \ldots, q_M\}$ denote the set of normalized qualities of all available targets. We denote the number of robots on the target $i \in \{1, \ldots, M\}$ by n_i, a nonnegative integer. The population fraction allocated to target i is $f_i = n_i/N$, which represents the target's relative frequency, and the vector of population fraction is $\mathbf{f} = [f_1, \ldots, f_M]^T$. The expected distribution is the set of desired population fractions for each task, $\mathbf{f^d} = [f_1^d, \ldots, f_M^d]^T$, where $f_i^d = q_i$. The usage of fractions rather than integers is practical for scaling, but it also introduces a distribution error as the fractions can take only certain values that are defined by the swarm size.

A relevant concept from set theory could be used to observe this as a set partitioning problem. A family X is a partition of a set E if and only if the elements of X are mutually disjoint and their union is E:

$$\begin{aligned} \bigcap_{x \in X} &= \emptyset \\ \bigcup_{x \in X} &= E. \end{aligned} \tag{7.2}$$

However, for the proposed scenario the system optimization based on the maximum utility cannot be applied because the combined utilities of the robots are unknown as robots have no knowledge of the decisions taken by other robots. Therefore, the DBA is proposed.

7.3.2 Distributed Bees Algorithm (DBA)

When a robot receives information about the targets it calculates the utilities with respect to those targets. The utility depends on the target's quality value and the related cost, i.e. the robot's distance from the target.

7.3.2.1 Costs

The cost of a target i for robot k is calculated as the Euclidean distance between the robot and the target in a two-dimensional arena:

$$d_i^k = \sqrt{(x_i - x_k)^2 + (y_i - y_k)^2}. \tag{7.3}$$

where (x_i, y_i) and (x_k, y_k) represent target's and robot's coordinates in the arena, respectively.

However, to calculate the utility the target's visibility is used defined as the reciprocal value of the distance:

$$\eta_i^k = \frac{1}{d_i^k}. \tag{7.4}$$

7.3.2.2 Qualities

The quality is a scalar value that represents priority, or the complexity of the target. Normalized qualities are calculated as fractions of the sum of qualities of all available targets:

$$q_i = \frac{Q_i}{\sum_{j=1}^{M} Q_j}. \tag{7.5}$$

where Q_i is a quality of the target i. In real-world scenarios, the quality of a region of interest is an estimated value that is as a result of sensor-readings or a previously acquired knowledge.

7.3.2.3 Utilities

The utility of a robot as proposed in (7.1) depends on both, cost and quality of the chosen target. The utility is defined as a probability that the robot k is allocated to the target i, and it is calculated as follows:

$$p_i^k = \frac{q_i^\alpha \eta_i^\beta}{\sum_{j=1}^{M} q_j^\alpha \eta_j^\beta} \tag{7.6}$$

where α and β are control parameters that allow us to bias the decision-making mechanism towards the quality of the solution or its cost, respectively. ($\alpha, \beta > 0$; $\alpha, \beta \in \Re$.) From (7.6) it is easy to show that

$$\sum_{i=1}^{M} p_i^k = 1. \tag{7.7}$$

7.3.2.4 Decision-making mechanism

The underlying decision-making mechanism of the DBA algorithm adopts the roulette rule, also known as the wheel-selection rule. That is, every target has an associated probability with which it is chosen from a set of available targets. Once all the probabilities are calculated as in (7.6), the robot will choose a target by "spinning the wheel".

It should be noticed that the resulting robots' distribution depends on their initial distribution in the arena, i.e. their distances from each target prior to target allocation. Therefore, robots' utilities will differ with respect to the same target if their distances from that target are not equal. Since a combined robots utility cannot be computed due to a distributed nature of the proposed algorithm, the quality of the targets is used as the only measure for the expected robots' distribution. Although the overall cost efficiency of the swarm is not analyzed here, target's visibility as used in (7.6) makes closer targets more attractive to robots.

7.4 Experimental Evaluation

In this section, the results from the experiments with real robots are presented to validate the performance of the proposed DBA. The overview of robots' hardware and the communication protocol used in experiments is given, followed by the description of the experimental setup. The experiments results are presented and discussed.

7.4.1 Robot hardware

The robots were assembled to have the same selection of hardware components (see Fig. 7.1). The Lynxmotion Terminator Sumo Robot Kit with four Spur Gear Head Motors was used as a base to build the robots, although any platform that could support the listed hardware components would be suitable. The DC motors were powered with 12 VDC, with 200 RPM, torque of 63.89 oz.in (4.6 Kg-cm), 30:1 reduction and 6 mm shaft diameter, and they were paired as left-hand and right-hand, in order to be able to perform rotation on-the-spot. Devantech MD22 Motor Driver was used to control the motors' rotation speed and direction. It averaged the PWM signal received from the Arduino microcontroller board to provide a proportional value of

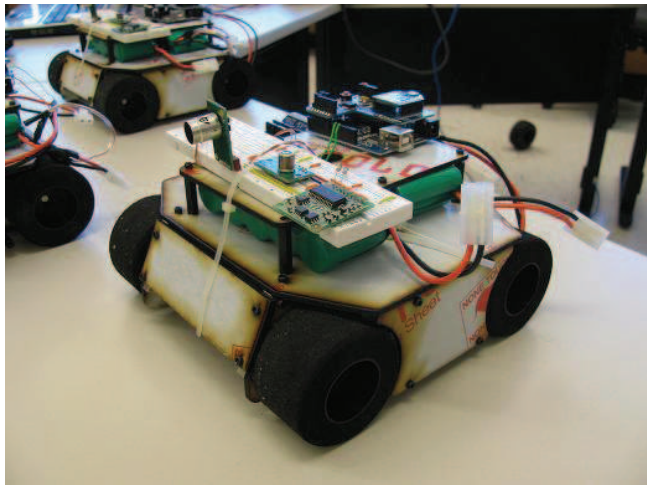


Figure 7.1: Sumo robot used in the experiments.

the 12 VDC from the battery. The four switches on the motor driver's board were used to define the working mode. In the experiments, the analog mode was used which provided satisfactory speed control.

Arduino Duemilanove microcontroller board with ATMEGA328 microcontroller was powered with the 6 VDC battery. ZigBee module used for robot communication was connected using the Arduino Xbee shield. Sensors were programmed for I^2C communication protocol with the microcontroller. One ultrasonic sensor was mounted on each robot for obstacle detection. Thermal sensor was used to detect the targets. The odometry error inherent to all mobile robots affected their precise localization. It cannot be eliminated but various methods for its reduction have been proposed, such as the averaging method proposed by Gutiérrez et al. (2009).

7.4.2 Coordinator and communication

A computer with a Pentium IV processor at 3 GHz with 2 GB of RAM was used to program the robots and to connect a ZigBee communication module that created a mesh communication network (the coordinator). The ZigBee modules mounted on robots were able to detect a reserved communication channel and connect with the coordinator. This allowed the communication between the robots and the robots with the computer. The broadcast mode that allowed each module to communicate with any other module in the network was used.

7.4.3 Experimental Results

The main objective of the experimental setup was to test the performance of the proposed algorithm and not the sensing and pattern recognition capabilities of the robots. Although the robots performed well in detecting the heat source and sending the estimated location and measured temperature, which was tested in initial experiments, in order to test the performance of the DBA algorithm a simplified scenario was arranged. A small swarm of three real and two simulated robots was used in search of two targets. The experiments were performed in 12×12 sq ft (3.65×3.65 m^2) arena, with randomly distributed obstacles. Robots were placed at the preprogrammed initial locations. When the command was sent from the coordinator the robots started the random search. After a certain period of time t_0, the information of two targets was sequentially sent from the coordinator. The information included the targets' estimated locations and their temperature (quality) values. The robots calculated the probabilities to move to each target as in 7.6. The real robots were not able to recognize the message sender; therefore, the coordinator was able to simulate two robots that found two different targets.

Two types of experiments were performed. In the first one, the random search time t_0 was changed to test its effect on the odometry error. Single robot was used in order to avoid the collision with other robots, and only one robot was simulated from the coordinator. With each of three robots 30 experiments were conducted in order to obtain the average odometry error value. The results of the first experimental setup are shown in Fig. 7.2. The search was considered successful if the robot was able to get as close as 30,48 cm (1 ft) from the target. It can be noticed from the Fig. 7.2 that while increasing the initial random search time of the robot, the odometry error increased as well. This happened due to the imperfect calibration of the DC motors, non-constant battery voltage, friction of the ground, etc. It can also be noticed that for the $t_0 < 90$ s the experiment success rate was 100 %.

The results from the first experimental setup were used to set the parameters for the second experimental setup. The random search initial time value was set to $t_0 = 30$ s, which guaranteed that the odometry error would be below the success threshold. The scenario involved all three robots in search for two targets whose information was sent form the coordinator. The choice of having two targets in the scenario was based on the number of real robots that were at our disposal. By having three robots and two targets we could test how the robots distributed in the arena based on the targets' quality values. Four possible events could occur: 1) all robots go for the first target (T1); 2) all robots go for the second target (T2); 3) two robots

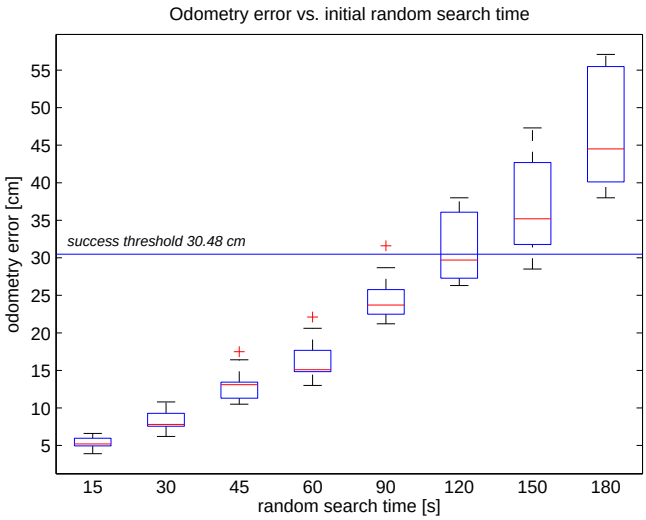


Figure 7.2: Results of the first experimental setup: Average odometry error vs. initial random search time.

go for the target T1 and one for the target T2; and 4) one robot goes for the target T1 and two robots go for the target T2.

Because of the relatively small size of the robot arena, the visibility of the targets was set to be constant, $h_1 = h_2 = 1$, and the target allocation was performed based only on the targets quality values. The values of the control parameters were also set to $\alpha = \beta = 1$. In order to test the self-organized behavior of the swarm of robots, the quality values of the two targets sent from the coordinator to the robots were changed. The experimental results are shown in Table 7.1. It can be notice that when the quality values were equal, $Q_1 = Q_2 = 50$, in most cases two robots would go for one target and one would go for the other. This was expected since the probabilities of choosing any of the targets were equal. By increasing the difference between the quality values, the distribution would change in favor of the target with the higher quality value because the probability that a robot chooses that target also increased.

The experimental results show that the task allocation was performed according to the targets' quality values in an autonomous and decentralized manner. The targets with higher quality values attracted more robots, which was the objective for the multi-foraging scenario. The odometry error inherent to mobile robots was used as

Table 7.1: Robots' distribution vs. targets' quality values

Quality values	T1:T2	Occurrence	Occurrence [%]
$Q1 = Q2 = 50$	2:1	11	36.67
$Q1 = Q2 = 50$	3:0	4	13.33
$Q1 = Q2 = 50$	1:2	13	43.33
$Q1 = Q2 = 50$	0:3	2	6.67
$Q1 = 70; Q2 = 30$	2:1	18	60.00
$Q1 = 70; Q2 = 30$	3:0	9	30.00
$Q1 = 70; Q2 = 30$	1:2	2	6.67
$Q1 = 70; Q2 = 30$	0:3	1	3.33
$Q1 = 90; Q2 = 10$	2:1	8	26.67
$Q1 = 90; Q2 = 10$	3:0	21	70.00
$Q1 = 90; Q2 = 10$	1:2	1	3.33
$Q1 = 90; Q2 = 10$	0:3	0	0.00

an advantage in order to gather the robots in the vicinity of the found targets and not at their exact locations. Still, there is a necessity to maintain the odometry error within the acceptable limits, and this is planned as a part of the future work.

7.5 Evaluation Through Simulations

The experiments with real robots could not provide the insight on the multirobot system's scalability because of the small number of available robots. Therefore, the experiments were performed in a simulated environment which provided the results for a statistical analysis of the algorithm's performance. In this section, the simulator and the simulation setup are described, and the simulation results are presented in order to analyze the scalability of the DBA.

7.5.1 Simulator

Our simulation platform is a fast, specialized multi-robot simulator for the e-puck robots described in (Gutiérrez et al., 2010). It is a simple and effective simulator implementing 2D kinematics. A screenshot of the simulator is shown in Fig. 7.3. In our simulations, the e-puck is modeled as a cylindrical body of 3.5 cm in radius that holds 8 infrared (IR) proximity sensors distributed around the body, 3 ground sensors

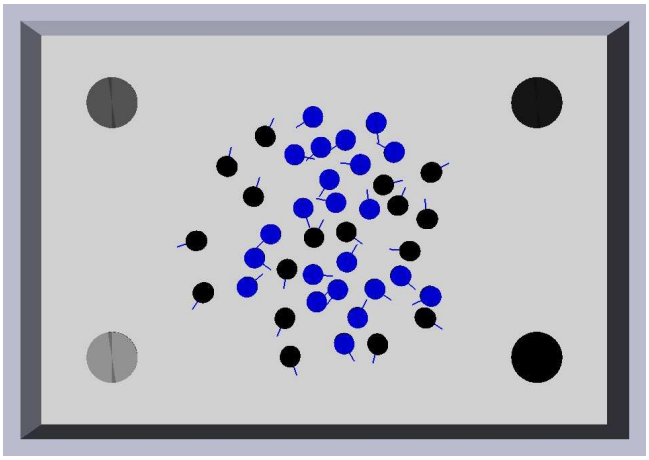


Figure 7.3: Simulator screenshot. Simulation setup included 40 robots engaged in search for 4 targets of different qualities represented by different grey-level intensity. Robots are programmed for obstacle avoidance; when robot detects an obstacle its color changes from black to blue to mark his new state. Once the robot has taken a new direction, its color goes back to black.

on the lower-front part of the body and a range and bearing communication sensor. IR proximity sensors have a range of 5 cm, while the communication range of the E-puck Range & Bearing module was set to cover the whole arena. For the three types of sensors, real robot measurements were sampled and the data was mapped into the simulator. Furthermore, uniformly distributed noise was added to the samples in order to effectively simulate different sensors; $\pm 20\%$ noise is added to the IR sensors and $\pm 30\%$ to the ground sensors. In the range and bearing sensor, noise is added to the range (± 2.5 cm) and bearing ($\pm 20°$) values. A differential drive system made up of two wheels is fixed to the body of the simulated robot. At each time step (100 ms), the robot senses the environment and actuates. The robot speed has been limited to 6 cm/s when moving straight and 3 cm/s when turning.

7.5.2 Simulation Setup

Three different simulation setups have been chosen to compare and study performance and scalability of the proposed DBA algorithm. The setups were carried out in the same arena where the number of robots, number of targets and targets' quality values were changed as shown in Table 7.2. Additional simulation setup was created in order

Table 7.2: Parameters describing three arenas used in simulations

	Arena 1					Arena 2					Arena 3				
Area dimensions $[m^2]$	1.5×2.125					1.5×2.125					1.5×2.125				
Number of robots	10	20	40	60	100	10	20	40	60	100	10	20	40	60	100
Simulation duration [time steps]	400	400	400	300	200	400	400	400	300	200	400	400	400	300	200
Time step duration $[s]$	0.1					0.1					0.1				
Initial area radius $[m]$	0.4	0.4	0.4	0.4	0.5	0.4	0.4	0.4	0.4	0.5	0.4	0.4	0.4	0.4	0.5
Number of targets	2					4					4				
Target radius $[m]$	0.09					0.09					0.09				
Target 1 location (x, y) $[m]$	(-0.45, 0.75)					(-0.45, 0.75)					(-0.45, 0.75)				
Target 2 location (x, y) $[m]$	(0.45, -0.75)					(0.45, -0.75)					(0.45, -0.75)				
Target 3 location (x, y) $[m]$	N/A					(-0.45, -0.75)					(-0.45, -0.75)				
Target 4 location (x, y) $[m]$	N/A					(0.45, 0.75)					(0.45, 0.75)				
Target 1 quality (q_1)	0.5					0.25					0.1				
Target 2 quality (q_2)	0.5					0.25					0.2				
Target 3 quality (q_3)	N/A					0.25					0.3				
Target 4 quality (q_4)	N/A					0.25					0.4				

* Targets have a form of a circle. Without loss of generality, their radius and location were intuitively chosen.

to analyze the effect of the control parameters α and β on the resulting distribution. Each simulation was repeated 50 times for different initial robot distribution in order to perform a statistical analysis of the results.

7.5.3 Simulation Results and Discussion

In order to test the scalability of the proposed DBA with respect to the size of the swarm, the simulations were performed with 10, 20, 40, 60 and 100 robots for the simulation setup 1, and 20, 40, 60 and 100 robots for the simulation setup 2 and the simulation setup 3. The number of targets was also changed, from two in the simulation setup 1 to four in the simulation setup 2, in order to test the performance of the algorithm with respect to the number of targets. In the simulation setup 3, four targets with different quality values were used in order to show the adaptability of the swarm to a non-uniform distribution of the "food" in the environment. This is also the most realistic scenario. Finally, the simulation setup 4 was created to test how the change in the ratio of the control parameters α and β can affect the resulting robots' distribution.

As the algorithm performance metrics the mean absolute error (MAE) of the robots' distribution is defined, which is given by

$$MAE = \frac{1}{M} \sum_{i=1}^{M} \left| f_i - f_i^d \right| \tag{7.8}$$

Table 7.3: Mean absolute error (MAE) of the robots' distribution

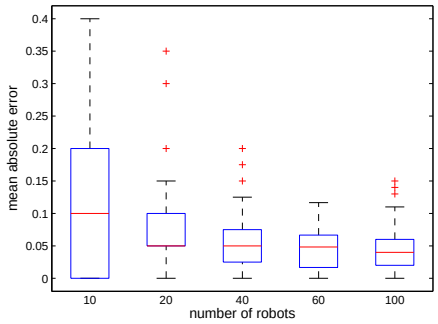
	Num. of robots	Average MAE	Maximum MAE
Sim. setup 1	10	0.1140	0.4000
	20	0.0820	0.3500
	40	0.0555	0.2000
	60	0.0482	0.1167
	100	0.0461	0.1100
Sim. setup 2	10	0.0941	0.1750
	20	0.0720	0.1500
	40	0.0500	0.1000
	60	0.0475	0.0917
	100	0.0313	0.0650
Sim. setup 3	10	0.0979	0.2500
	20	0.0790	0.1500
	40	0.0526	0.0875
	60	0.0478	0.0790
	100	0.0343	0.0750

* Parameters for each simulation setup are described in Table 7.2.
** The average MAE and the maximum MAE values were obtained from 50 simulations performed for the each swarm size within the each simulation setup.

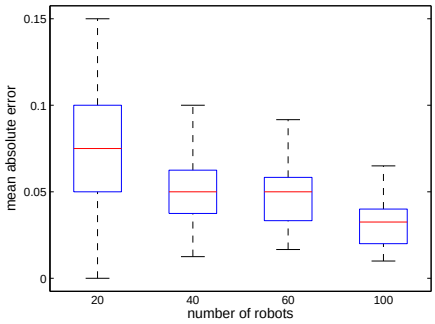
where $f_i^d = q_i$.

As the name suggests, the mean absolute error is the average value of the absolute distribution error (per target) that is the result of discrepancy between the expected and the resulting robots' distribution. For each simulation setup and each swarm size described in Table 7.2 fifty simulations were performed. The average and the maximum values of MAE obtained from the simulations are presented in Table 7.3 and graphically shown in Fig. 7.4. It can be noticed that the average MAE and maximum MAE values decrease as the size of the robot swarm increases regardless of the number of targets or their quality values. This was expected because of the probabilistic target allocation mechanism applied in (7.6).

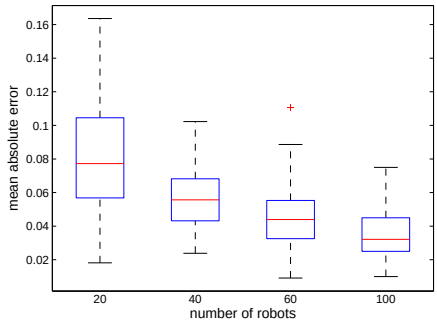
The effectiveness of the algorithm in terms of increased number of targets can be seen from the results shown in Fig. 7.5 and Fig. 7.6. The results show that the



(a) Two equal targets



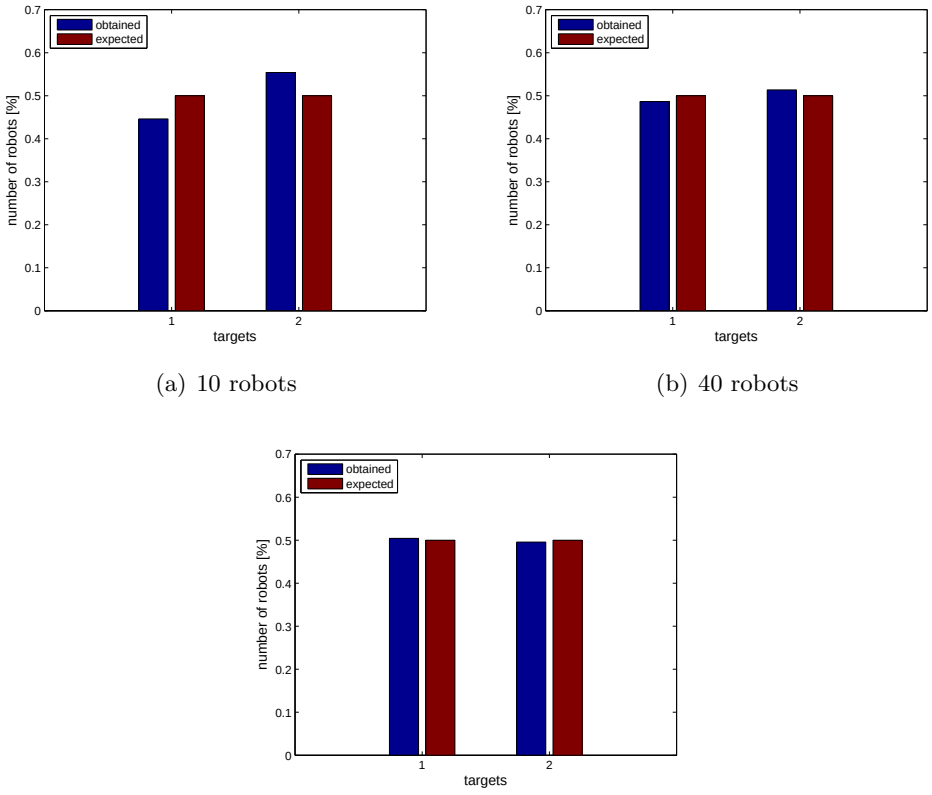
(b) Four equal targets



(c) Four different targets

Figure 7.4: Box-plot comparison shows the robots' distribution mean absolute error (MAE) with respect to the swarm size: a) simulation setup 1; b) simulation setup 2; and c) simulation setup 3. Each box-plot comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown with a plus symbol. The values were obtained from 50 simulations performed for each swarm size within each simulation setup.

average and the maximum MAE values decreased for larger swarms in case of 4 targets of the same quality. It should be noticed that the allocation of 10 robots to 4 targets produces an error that is the result of the cardinality of the robot swarm. It is not physically possible to partition the swarm in order to obtain the expected



(a) 10 robots

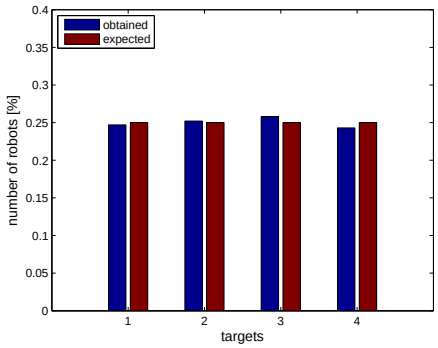
(b) 40 robots

(c) 100 robots

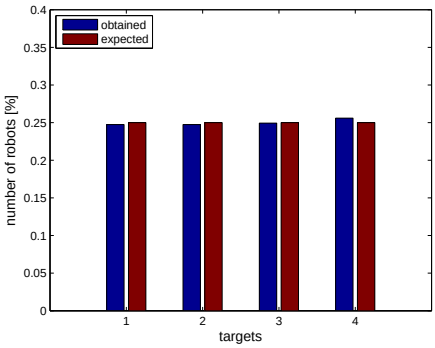
Figure 7.5: Bar-plot comparison of the expected (red) vs the obtained (blue) robots' distribution on two targets of same quality values, $q_1 = q_2 = 0.5$. Fifty simulations were performed for each of the following swarm sizes: a) 10 robots; b) 40 robots; and d) 100 robots.

target allocation (2.5 robots per target).

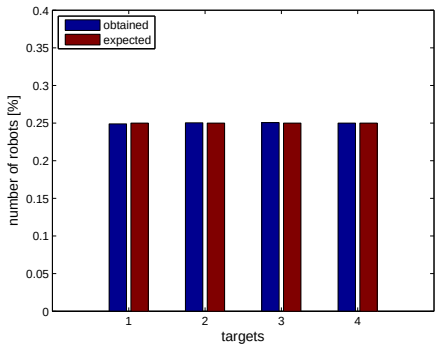
Another inherent source of error results from the assumption that the robots that had found a target are not allowed to reallocate to another target, therefore they are not involved in the decision-making process. Also, it is assumed that the robots wait for a predetermined number of targets to be found before they make a decision, which can result in the same target being found by more than one robot. This fraction of the robot swarm also produces an error in the final distribution because they cannot reallocate to another target. The algorithm's performance is analyzed having these



(a) 20 robots



(b) 60 robots

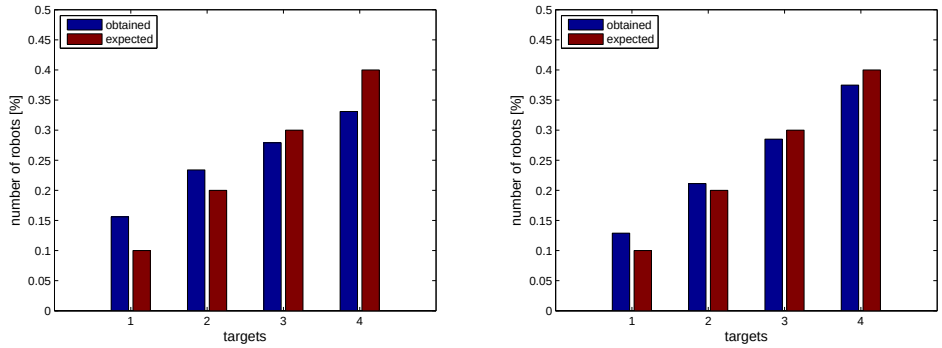


(c) 100 robots

Figure 7.6: Bar-plot comparison of the expected (red) vs the obtained (blue) robots' distribution on four targets of same quality values, $q_1 = q_2 = q_3 = q_4 = 0.25$. Fifty simulations were performed for each of the following swarm sizes: a) 20 robots; b) 60 robots; and d) 100 robots.

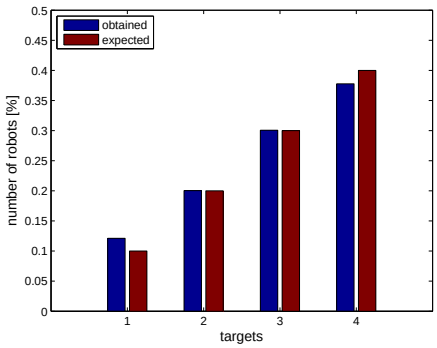
issues mind.

In order to test the ability of the robot swarm to adapt to a non-uniform distribution of "food" in the environment, the simulation were performed for four different targets (simulation setup 3). The robots' distribution changed according to a new set of targets' quality values, as shown in Fig. 7.7. It can also be noticed in the same figure that the resulting robots' distribution, with respect to the expected distribution, is slightly in favor of the less valuable targets. This is another consequence of the robots that had found a target not being able to reallocate, and it is especially



(a) 20 robots

(b) 60 robots



(c) 100 robots

Figure 7.7: Bar-plot comparison of the expected (red) vs the obtained (blue) robots' distribution on four targets of different quality values, $q_1 = 0.1$, $q_2 = 0.2$, $q_3 = 0.3$, and $q_4 = 0.4$. Fifty simulations were performed for each of the following swarm sizes: a) 20 robots; b) 60 robots; and d) 100 robots.

evident for smaller robot swarms. For example, let's consider a swarm of 10 robots in search of 4 different targets, as shown in Fig. 7.7(a). If in the random target search process two robots find the target with the associated quality value of 0.1, then the final relative frequency for this target cannot be less than 0.2 (2 out of 10 robots) which is already above the expected value of 0.1. Although for the larger swarms the effect of the initial robot distribution becomes less relevant, it is always present.

The control parameters, α and β, were introduced in (7.6) to compensate for the biased distribution, but also to allow us to give more relevance to either the quality

Table 7.4: Effects of control parameters on robots' distribution

α/β ratio	Average MAE	Maximum MAE
1	0.0478	0.1083
2	0.0525	0.1000
5	0.1415	0.2083

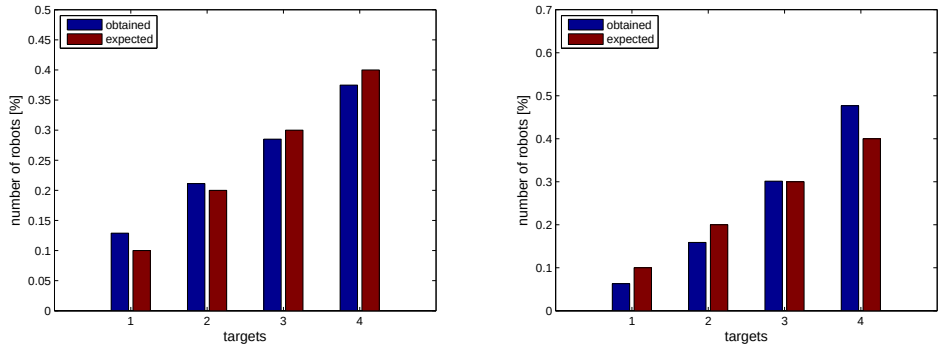
* The values were obtained from 50 simulation performed on the swarm of 60 robots in search for four targets with different quality values (simulation setup 3).

of the targets or the cost of reaching them. Therefore, in the simulation setup 4 the α/β ratio was increased to give more relevance to the quality value of the targets on the expense of their distances from the robots. The resulting robots' distributions per target for different values of the α/β ratio are presented in Fig. 7.8. Results show that, by tuning the control parameters, the final robot distribution can change in favor of the more valuable targets but with an increase in the average MAE (see Table 7.4). It is reasonable to expect that by decreasing the α/β ratio the cost efficiency of the robot swarm would improve in terms of the distance traveled, however, the MAE is also expected to increase. Further analysis of the effect of the control parameters will be a part of the future work.

7.6 Summary

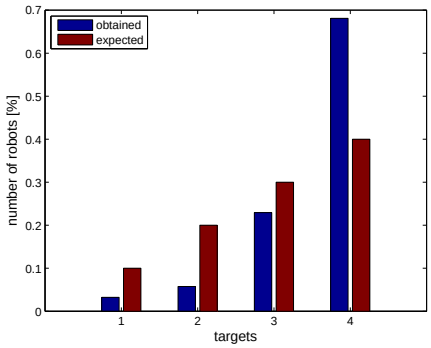
Various applications for large multirobot systems require efficient task allocation in terms of individual and combined robots' utilities. The quality of the solution is analyzed using a defined performance metrics, which in our case was a mean absolute error of the resulting robots' distribution with respect to the qualities of the available targets in the robot arena. In case of large, autonomous, multirobot systems, the scalability and the ability to adapt to different environments are the features of utmost importance. Our experiments through simulation showed that the proposed Distributed Bees Algorithm provides the robot swarm with scalability in terms of the number of robots and number of targets, but also with adaptability to a non-uniform distribution of the targets' qualities.

The importance of the control parameters, α and β, is that they provide a mechanism to adjust the robot swarm behavior depending on the task at hand and the available resources. In this chapter, the values of α and β were changed in order



(a) $\alpha/\beta = 1$

(b) $\alpha/\beta = 2$



(c) $\alpha/\beta = 5$

Figure 7.8: Effects of the control parameters, α and β, on the final robots' distribution. Target allocation was performed with 60 robots as described in the simulation setup 3 consisting of 4 targets with different quality values: $q_1 = 0.1$, $q_2 = 0.2$, $q_3 = 0.3$, and $q_4 = 0.4$. The results of the robots' distribution per target are shown for the following values of α/β ratio: a) $\alpha/\beta = 1$; b) $\alpha/\beta = 2$; c) $\alpha/\beta = 5$. The values were obtained from 50 simulations for each scenario.

to bias the resulting robots' distribution towards the more favorable targets. Future work will include the analysis of the effect of these parameters in terms of the task-allocation cost with respect to the distance traveled by the robots.

Part IV

Conclusions and Future Work

Chapter 8

Conclusions

Swarm Intelligence is a useful tool for solving a number of real-world problems. The decentralized approach in the design of multi-agent systems offers many advantages such as greater autonomy, scalability, robustness, and adaptability to a dynamically changing environment. In addition, this approach allows the development of Swarm Intelligence tools that can be used for optimization and feature extraction.

Swarming behavior in nature is a result of the evolution process, and provides us with an important advantage in solving complex problems, because through this behavior animal swarms have been able to withstand the challenges of nature, which was crucial for their survival. The foraging of ant and bee colonies inspired many algorithms whose initial purpose was to solve combinatorial optimization problems. The emergent properties of the swarming behavior were later used for a new problem-solving approach in applications of function minimization, network routing, image processing, data clustering, etc. Although the biological plausibility of Swarm Intelligence algorithms can be a matter of study from the biologists' point of view, in computer science and engineering, the researchers are guided by efficiency, flexibility, robustness and cost as possible criteria.

In this dissertation, we proposed a general design methodology for Swarm Intelligence tools. The proposed methodology was implemented in several application domains, which serve as illustrative examples. The methodology assumes a discrete data space defined by nodes on which the agents in the swarm move. The underlying decision-making mechanism adopts the roulette-wheel selection rule. That is, every node has an associated probability with which it is chosen from a set of available nodes. The probability values are calculated from the application-specific variables. Each agent autonomously calculates the probabilities of displacement to each node

in its neighborhood and chooses the next node by "spinning the wheel". By moving from node to node, the agents alter the environment that surrounds them, and the global patterns that emerge as the result of this behavior represent solutions to the given problem.

In this dissertation, the research in Swarm Intelligence focused on two general approaches. One approach was to solve the optimization-like problems using the swarm-based algorithms as tools, and the other approach was to model the multi-agent systems such that they resemble swarms of animals in nature providing them with the ability to autonomously perform a task at hand. This line of research has explored various applications for Swarm Intelligence principles, such as combinatorial optimization, image processing, data clustering, and distributed task allocation for multi-agent systems.

8.1 Summary of Contributions

The original contributions are made in the areas of design methodology and application of Swarm Intelligence tools. More specifically, the contributions of this dissertation are the following:

- **To propose a general design methodology for Swarm Intelligence tools.**

  We proposed a unified probabilistic decision-making rule for local interaction as a general design methodology for Swarm Intelligence tools in various application domains. The methodology requires the definition of a discrete data space in which the agents move, the set of application-specific variables, and the objective function for solutions evaluation.

- **To validate the proposed methodology in a real-world scenario.**

  The methodology was validated on the scenario of Unmanned Aerial Vehicle path optimization, treated as a combinatorial optimization problem. A colony of artificial ants was used to successfully find the optimal path for a set of the given waypoints, and the proposed method improved the system's performance in terms of time and resources needed for the UAV's flight. (Jevtić et al., 2010a).

Further, based on the proposed general methodology, we developed a set of novel tools that can be seen as illustrative examples of its implementation. These tools are the following:

- **Novel Swarm Intelligence method for adaptive edge detection in digital images.**

  The proposed edge detector was inspired by the foraging behavior of ant colonies, where the discrete space in which the ants move is defined by pixels in digital image. The pixels are featured by the gray-level intensity value. The artificial ants search for the pixels where distinct intensity changes occur, and the amount of the pheromone deposit that they lay on their path reflects the detected change in contrast. The accumulated pheromone trails attract other ants in the swarm to follow the marked route, and the emerging pheromone structure is a result of the cooperative behavior and indirect communication via environment. The accumulated pheromone represents the found true edges in the image. The images were preprocessed using the Multiscale Adaptive Gain for nonlinear contrast enhancement. (Jevtić et al., 2009b).

  The proposed method includes several parameters that are selected for efficient edge detection, and their further optimization could improve the overall performance. It was demonstrated through experiments that the proposed method outperforms other state-of-the-art ant colony-based edge detectors. In comparison to the conventional edge detectors that were not based on Swarm Intelligence, the proposed method has a higher computational cost. The analysis of the algorithm's adaptability was performed on a dynamically changing set of images. The algorithm successfully adapts to these changes, which is reflected in emergence of different global pheromone patterns. The adaptive behavior of artificial ants colonies can be applied to any type of digital habitat, which can lead to a new set of applications. (Jevtić and Andina, 2010).

- **Novel Swarm Intelligence method for broken-edge linking in digital images.**

  The proposed method applies the path optimization concept inherent to ant colonies in finding the optimal solutions for missing edge segments. This method can serve as a complementary tool to any edge detector. One of novelties of the proposed method was the introduction of the grayscale visibility matrix to define the fitness function for the evaluation of the found edge segments. The favorable solutions were the edge segments that contained less pixels and had higher mean and lower standard deviation of the pixels' grayscale visibility. The advantage of the proposed method was in applying the grayscale visibility matrix as the initial pheromone trail matrix which increased the probability

that the pixels belonging to the true edges are being chosen by the ants on their initial routes, which reduced the computational load. The effectiveness of the proposed method was demonstrated on a set of different test images and the experimental results confirmed its excellent performance. (Jevtić et al., 2009a).

- **Novel Swarm Intelligence method for cluster analysis.**

  The proposed Ant System-based Clustering Algorithm (ASCA) applies the pheromone-laying and pheromone-following concept of ant colonies to create a pheromone map in the data space. The highest concentration of pheromone deposits are found around clusters centers. Therefore, the proposed method can be applied to extract the number of clusters from a data set, and is also an efficient tool for detecting groups of atypical data patterns. The method was applied as an image-segmentation tool, specifically for detection of small, atypical groups of pixels. This represents a very useful feature for a variety of computer vision applications such as digital mammography, detection of industrial design imperfections, etc. In the experiments, the proposed method outperformed the state-of-the-art algorithms such as 1D-SOM, k-Means, FCM and PFCM. (Jevtić et al., 2011b).

- **Novel Swarm Intelligence method for distributed task allocation in multi-agent systems.**

  The proposed Distributed Bees Algorithm (DBA) was applied to the problem of target allocation in a swarm of robots. The expected distribution of the robots is obtained from the targets' quality values, which represent how rewarding the given targets are. The solution was evaluated using a defined performance metrics, which in our case was the mean absolute error of the resulting robots' distribution with respect to their expected distribution. The algorithm was validated through experiments with real robots. The odometry error of the mobile robots was used as an advantage to group the robots around the found targets. The experimental results were used to obtain the time threshold for scenario execution, in order to maintain the odometry error within the predefined limit. (Jevtić et al., 2010b).

  The simulator was developed to test the scalability of the proposed method. Our experiments showed that the proposed DBA provides the robot swarm with scalability in terms of the number of robots and number of targets, but also with adaptability to a non-uniform distribution of the targets' qualities. Two

parameters were used to control the importance level of the targets' distances and qualities. They provide a mechanism to adjust the robot swarm behavior depending on the task at hand and the available resources. (Jevtić et al., 2011a).

8.2 Future Work

With the experience gained from the work on various applications of Swarm Intelligence, the following lines of future research work are proposed:

- Optimization and automatic detection of the parameters of the proposed edge detection method for improved image segmentation results. Until now these parameters were experimentally obtained.
- Extension of the work on the edge detection algorithm's adaptability in order to apply it to other digital habitats. The algorithm optimization for faster execution would make it suitable for real-time image processing.
- The parameters of the proposed broken-edge linking method were experimentally obtained. The parameters' optimization, and the evaluation of the found edge segments with a novel fitness function, could be applied for improved edge detection.
- Combination of the edge detector with the broken-edge linking algorithm in order to develop a more exhaustive edge-detection method. Further optimization of the hybrid algorithm's parameters should also provide a better performance in terms of speed and edges' quality. Modification of the two methods, and the hybrid method, for edge detection in color images can open a wide range of new applications.
- The proposed data-clustering algorithm is an initial variant and leaves space for improvements. Future work will include the optimization of the algorithm's parameters in order to improve the adaptability of the algorithm to different data sets. For the task of image processing, more features such as spatial information or texture can be used to obtain better segmentation results. Also, testing the algorithm on the color images may open a wide range of computer vision applications.
- Testing of the proposed Distributed Bees Algorithm on a large number of robots in a real-world scenario such as exploration, search and rescue, etc. Modifica-

tions of the algorithm would be required to adjust to different types of communication between the robots, such as ZigBee, Infrared, field markers, etc.

- Extension of the work on the adaptability of large and complex multi-agent systems, and predictability of their behavior, in order to develop a paradigm for collaborative robotic swarms. The following objectives should be accomplished: i) to create a theoretical framework for the emergent systems that are based on the rules of local interactions which are associated as instances from swarms' level goals and tasks of interaction and collaboration; ii) to provide verification tools and methodology to formally specify and verify robustness, dependability, adaptability, manageability, and safety of the swarms; iii) to create a virtual prototype to test the framework, tools and methodology, and manageability and other activities of cognitive systems in virtual environment; iv) to develop, implement and test an integrated, adaptive robotic system, a cognitive system which is made of multiple heterogeneous agents of different capabilities and grouped in swarms.

So far, this line of research was carried out in the domain of computer science and engineering, but it supports applications that go beyond that as the modeling of multi-agent systems can be carried over to the domains of biology, medicine, sociology, economy, business, etc. The processes in nature and many processes in human society show emergent properties that are the result of multiple interactions between large numbers of individuals. Many scientific disciplines try different approaches in order to describe this phenomenon. By determining the relation between the stochastic processes on a lower level and the organized complexity on the system's global level, the predictability of such systems could be improved. This would have a high impact in the above-mentioned application domains, naming for example the economy and the predictability of the market. But also, the research on the models of Computational Swarm Intelligence can provide important feedback for the study of the natural swarms from which they were inspired.

Bibliography

Afshar, A., Bozorg Haddad, O., Mariño, M. A., and Adams, B. J. (2007). Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation. *Journal of the Franklin Institute*, 344(5):452–462.

Alarcón-Mondéjar, M. J., Zorzano-Mier, F. J., Jevtić, A., and Andina, D. (2008). Telecommunications network planning and maintenance. In *Proceedings of the 12th World Multi-Conference on Systemics, Cybernetics and Informatics, WMSCI 2008*, volume 8, pages 64–68.

Andina, D. and Jevtić, A. (2007a). Comparison results between usual backpropagation and modified backpropagation with weighting. In *Proceedings of the 11th WSEAS CSCC 2007: Systems Theory and Applications*, volume 2, pages 243–247.

Andina, D. and Jevtić, A. (2007b). Improved multilayer perceptron design by weighted learning. In *Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE 2007*, pages 3424–3429.

Andina, D., Jevtić, A., Marcano-Cedeño, A., and Barrón-Adame, J. M. (2007). Error weighting in artificial neural networks learning interpreted as a metaplasticity model. In Mira, J. and Álvarez, J., editors, *Bio-inspired Modeling of Cognitive Tasks*, volume 4527 of *Lecture Notes in Computer Science*, pages 244–252. Springer Berlin / Heidelberg.

Andina, D. and Pham, D. T. (2007). *Computational Intelligence: For Engineering and Manufacturing*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923.

Awad, M., Chehdi, K., and Nasri, A. (2009). Multi-component image segmentation using a hybrid dynamic genetic algorithm and fuzzy c-means. *IET Image Processing*, 3(2):52–62.

Bailis, P., Nagpal, R., and Werfel, J. (2010). Positional communication and private information in honeybee foraging models. In Dorigo, M., Birattari, M., Di Caro, G., Doursat, R., Engelbrecht, A., Floreano, D., Gambardella, L., Groß, R., Sahin, E., Sayama, H., and Stützle, T., editors, *Swarm Intelligence*, volume 6234 of *Lecture Notes in Computer Science*, pages 263–274. Springer Berlin / Heidelberg.

Barrón-Adame, J. M., Herrera-Delgado, J. A., Cortina-Januchs, M. G., Andina, D., and Vega-Corona, A. (2007). Air pollutant level estimation applying a self-organizing neural network. In Mira, J. and Álvarez, J., editors, *Nature Inspired Problem-Solving Methods in Knowledge Engineering*, volume 4528 of *Lecture Notes in Computer Science*, pages 599–607. Springer Berlin / Heidelberg.

Beni, G. and Wang, J. (1989). Swarm intelligence in cellular robotic systems. In *Proceedings of the NATO Advanced Workshop on Robotics and Biological Systems*, Il Ciocco, Tuscany, Italy.

Berman, S., Halasz, A., Hsieh, M. A., and Kumar, V. (2009). Optimized stochastic policies for task allocation in swarms of robots. *IEEE Transactions on Robotics*, 25(4):927–937.

Berman, S., Halasz, A., Kumar, V., and Pratt, S. (2007). Bio-inspired group behaviors for the deployment of a swarm of robots to multiple destinations. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pages 2318–2323.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA.

Blum, C. and Dorigo, M. (2004). The hyper-cube framework for Ant Colony Optimization. *IEEE Transactions on Systems, Man & Cybernetics, Part B: Cybernetics*, 34(2):1161–1172.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems*. Oxford University Press, Inc., New York, NY, USA.

Bonomi, E. and Lutton, J.-L. (1984). The n-city travelling salesman problem: Statistical mechanics and the metropolis algorithm. *SIAM Review*, 26(4):551–568.

Breder, C. M. (1954). Equations descriptive of fish schools and other animal aggregations. *Ecology*, 35(3):361–370.

Bullnheimer, B., Hartl, R. F., and Strauß, C. (1999). A new rank based version of the Ant System: A computational study. *Central European Journal for Operations Research & Economics*, 7(1):25–38.

Burgard, W., Moors, M., Stachniss, C., and Schneider, F. E. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–386.

Cai, W., Chen, S., and Zhang, D. (2007). Fast and robust fuzzy c-means clustering algorithms incorporating local information for image segmentation. *Pattern Recognition*, 40(3):825–838.

Camazine, S., Franks, N. R., Sneyd, J., Bonabeau, E., Deneubourg, J.-L., and Theraulaz, G. (2001). *Self-Organization in Biological Systems*. Princeton University Press, Princeton, NJ, USA.

Campo, A. and Dorigo, M. (2007). Efficient multi-foraging in swarm robotics. In *Proceedings of the 9th European Conference on Advances in Artificial Life. ECAL'07*, pages 696–705, Berlin, Heidelberg. Springer-Verlag.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 8:679–714.

Chang, K.-C. and Yeh, M.-F. (2005). Grey relational analysis based approach for data clustering. *IEE Proceedings - Vision, Image & Signal Processing*, 152(2):165–172.

Chu, S.-C., Roddick, J. F., Su, C.-J., and Pan, J.-S. (2004). Constrained ant colony optimization for data clustering. In Zhang, C., Guesgen, H. W., and Yeap, W. K., editors, *PRICAI 2004: Trends in Artificial Intelligence*, volume 3157 of *Lecture Notes in Computer Science*, pages 534–543. Springer Berlin / Heidelberg.

Chuang, K. S., Tzeng, H. L., Chen, S., Wu, J., and Chen, T. J. (2006). Fuzzy c-means clustering with spatial information for image segmentation. *Computerized Medical Imaging & Graphics*, 30(1):9–15.

Cordon, O., Fernandez de Viana, I., Herrera, F., and Moreno, L. (2000). A new ACO model integrating evolutionary computation concepts: The best-worst Ant System. In Dorigo, M., Middendorf, M., and Sttzle, T., editors, *Abstract Proceedings of*

ANTS'2000–From Ant Colonies to Artificial Ants, International Workshops on Ant Algorithms, pages 22–29. Université Libre de Bruxelles, Belgium.

Cortina-Januchs, M. G., Quintanilla-Domínguez, J., Jevtić, A., and Andina, D. (2008). Telecommunications network planning and maintenance. In *Proceedings of the 12th World Multi-Conference on Systemics, Cybernetics and Informatics, WMSCI 2008*, volume 8, pages 74–79.

Crespi, V., Galstyan, A., and Lerman, K. (2008). Top-down vs bottom-up methodologies in multi-agent system design. *Autonomous Robots*, 24(3):303–313.

Danahy, E. E., Panetta, K. A., and Agaian, S. S. (2007). Coordinate logic transforms and their use in the detection of edges within binary and grayscale images. In *IEEE International Conference on Image Processing, 2007. ICIP 2007.*, volume 3, pages III–53–III–56.

Deneubourg, J. L., Aron, S., Goss, S., and Pasteels, J. M. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior*, 3(2):159–168.

Deneubourg, J. L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., and Chrétien, L. (1991). The dynamics of collective sorting robot-like ants and ant-like robots. In *Proceedings of the 1st International Conference on Simulation of Adaptive Behavior on From Animals to Animats*, pages 356–363, Cambridge, MA, USA. MIT Press.

Dias, M. B., Zlot, R., Kalra, N., and Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270.

Doctor, S., Venayagamoorthy, G. K., and Gudise, V. G. (2004). Optimal PSO for collective robotic search applications. In *Proceedings of the 2004 Congress on Evolutionary Computation, CEC2004*, volume 2, pages 1390–1395.

Dorigo, M. and Blum, C. (2005). Ant Colony Optimization theory: A survey. *Theoretical Computer Science*, 344:243–278.

Dorigo, M. and Gambardella, L. M. (1997). Ant Colony System: A cooperating learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computing*, 1(1):53–66.

Dorigo, M., Maniezzo, V., and Colorni, A. (1996). Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man & Cybernetics, Part B: Cybernetics*, 26(1):29–41.

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*. MIT Press, Cambridge, MA, USA.

Duan, H.-b., Zhang, X.-y., Wu, J., and Ma, G.-j. (2009). Max-min adaptive ant colony optimization approach to multi-UAVs coordinated trajectory replanning in dynamic and uncertain environments. *Journal of Bionic Engineering*, 6(2):161–173.

Dudek, G., Jenkin, M., and Milios, E. (2002). A taxonomy of multirobot systems. In Balch, T. and Parker, L., editors, *Robot Teams: From Diversity to Polymorphism*, pages 3–22. A.K. Peters, Natick, Massachusetts.

Eberhart, R. C. and Shi, Y. (2000). Comparing inertia weights and constriction factors in Particle Swarm Optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 84–88.

Engelbrecht, A. P. (2005). *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons, Ltd, Chichester, UK.

Farinelli, A., Iocchi, L., Nardi, D., and Ziparo, V. A. (2006). Assignment of dynamically perceived tasks by token passing in multirobot systems. *Proceedings of the IEEE*, 94(7):1271–1288.

Franchi, A., Freda, L., Oriolo, G., and Vendittelli, M. (2009). The sensor-based random graph method for cooperative robot exploration. *IEEE/ASME Transactions on Mechatronics*, 14(2):163–175.

Gambardella, L. M. and Dorigo, M. (1995). Ant-Q: A reinforcement learning approach to the traveling salesman problem. In Prieditis, A. and Russell, S., editors, *Proceedings of the 12th International Conference on Machine Learning*, pages 252–260. Morgan Kaufmann Publishers.

Gerkey, B. P. and Matarić, M. J. (2004). A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, 23(9):939–954.

Golden, B. L. and Skiscim, C. C. (1986). Using simulated annealing to solve routing and location problems. *Naval Research Logistics Quarterly*, 33(2):261–279.

Gomez, J. and Jamshidi, M. (2010). Fuzzy adaptive control for a UAV. *Journal of Intelligent & Robotic Systems*, pages 1–23.

Gonzalez, R. C. and Woods, R. E. (2008). *Digital Image Processing*. Prentice Hall, New Jersey, USA, 3rd edition.

Grau, J. B., Anton, J. M., Packianather, M. S., Ermolov, I., Aphanasiev, R., Cisneros, J. M., Cortina-Januchs, M. G., Jevtić, A., and Andina, D. (2009). Sustainable agriculture using an intelligent mechatronic system. In *Proceedings of the 35th Annual Conference of IEEE Industrial Electronics, IECON '09*, pages 3416–3421.

Grefenstette, J. J., Gopal, R., Rosmaita, B. J., and Gucht, D. V. (1985). Genetic algorithms for the traveling salesman problem. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 160–168, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

Groß, R., Nouyan, S., Bonani, M., Mondada, F., and Dorigo, M. (2008). Division of labour in self-organised groups. In *Proceedings of the 10th international conference on Simulation of Adaptive Behavior: From Animals to Animats*, SAB '08, pages 426–436, Berlin, Heidelberg. Springer-Verlag.

Gutiérrez, A., Campo, A., Monasterio-Huelin, F., Magdalena, L., and Dorigo, M. (2010). Collective decision-making based on social odometry. *Neural Computing & Applications*, 19(6):807–823.

Gutiérrez, A., Campo, A., Santos, F. C., Monasterio-Huelin, F., and Dorigo, M. (2009). Social odometry: Imitation based odometry in collective robotics. *International Journal of Advanced Robotic Systems*, 26(2):129–136.

Hamilton, W. D. (1971). Geometry for the selfish herd. *Journal of Theoretical Biology*, 31(2):295–311.

Handl, J., Knowles, J., and Dorigo, M. (2006). Ant-based clustering and topographic mapping. *Artificial Life*, 12(1):35–61.

Handl, J. and Meyer, B. (2002). Improved ant-based clustering and sorting in a document retrieval interface. In Guervós, J., Adamidis, P., Beyer, H.-G., Schwefel, H.-P., and Fernández-Villaca˜as, J.-L., editors, *Parallel Problem Solving from Nature – PPSN VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 913–923. Springer Berlin / Heidelberg.

Handl, J. and Meyer, B. (2007). Ant-based and swarm-based clustering. *Swarm Intelligence*, 1(2):95–113.

He, X., Jia, W., Hur, N., Wu, Q., Kim, J., and Hintz, T. (2006). Bilateral edge detection on a virtual hexagonal structure. In Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Nefian, A., Meenakshisundaram, G., Pascucci, V., Zara, J., Molineros, J., Theisel, H., and Malzbender, T., editors, *Advances in Visual Computing*, volume 4292 of *Lecture Notes in Computer Science*, pages 176–185. Springer Berlin / Heidelberg.

Huang, P., Cao, H., and Luo, S. (2008). An artificial ant colonies approach to medical image segmentation. *Computer Methods & Programs in Biomedicine*, 92:267–273.

Jaimes, A. and Jamshidi, M. (2010). Consensus-based and network control of UAVs. In *Proceedings of the 5th IEEE International Conference on System of Systems Engineering (SoSE 2010)*, pages 1–6.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323.

Jamshidi, M. (2009). *System of Systems Engineering - Innovations for the 21st Century*. John Wiley & Sons, New York, NY, USA.

Jevtić, A. and Andina, D. (2010). Adaptive artificial ant colonies for edge detection in digital images. In *Proceedings of the 36th Annual Conference on IEEE Industrial Electronics Society, IECON 2010*, pages 2813–2816.

Jevtić, A., Andina, D., Jaimes, A., Gomez, J., and Jamshidi, M. (2010a). Unmanned aerial vehicle route optimization using Ant System algorithm. In *Proceedings of the 5th IEEE International Conference on System of Systems Engineering, SoSE 2010*, pages 1–6.

Jevtić, A., Gazi, P., Andina, D., and Jamshidi, M. (2010b). Building a swarm of robotic bees. In *2010 World Automation Congress, WAC 2010*, pages 1–6.

Jevtić, A., Gutiérrez-Martín, A., Andina, D., and Jamshidi, M. (2011a). Distributed Bees Algorithm for task allocation in swarm of robots. *IEEE Systems Journal, In preparation.*

Jevtić, A., Melgar, I., and Andina, D. (2009a). Ant based edge linking algorithm. In *Proceedings of the 35th Annual Conference of the IEEE Industrial Electronics Society, IECON 2009*, pages 3353–3358.

Jevtić, A., Quintanilla-Domínguez, J., Barrón-Adame, J. M., and Andina, D. (2011b). Image segmentation using Ant System-based Clustering Algorithm. In *International Conference on Soft Computing Models in Industrial & Environmental Applications, SOCO 2011*. Accepted for publication.

Jevtić, A., Quintanilla-Domínguez, J., Cortina-Januchs, M. G., and Andina, D. (2009b). Edge detection using ant colony search algorithm and multiscale contrast enhancement. In *Proceedings of the 2009 IEEE International Conference on Systems, Man & Cybernetics, SMC 2009*, pages 2193–2198.

Jiang, J. A., Chuang, C. L., Lu, Y. L., and Fahn, C. S. (2007). Mathematical-morphology-based edge detectors for detection of thin edges in low-contrast regions. *IET Image Processing*, 1(3):269–277.

Jiang, Y. and Zhou, Z.-H. (2004). SOM ensemble-based image segmentation. *Neural Processing Letters*, 20:171–178.

Johnson, D. S. and McGeoch, L. A. (1997). The traveling salesman problem: A case study in local optimization. In Aarts, E. H. L. and Lenstra, J. K., editors, *Local Search in Combinatorial Optimization*, pages 215–310. John Wiley and Sons Ltd., Chichester, UK.

Joordens, M. A. and Jamshidi, M. (2010). Consensus control for a system of underwater swarm robots. *IEEE Systems Journal*, 4(1):65–73.

Junwei, T. and Yongxuan, H. (2007). Histogram constraint based fast FCM cluster image segmentation. In *Proceedings of the IEEE International Symposium on Industrial Electronics, 2007. ISIE 2007.*, pages 1623–1627.

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, A. Y. (2002). An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 24(7):881–892.

Karaboga, D. and Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation*, 214(1):108–132.

Keller, E. F. and Segel, L. A. (1970). Initiation of slime mold aggregation viewed as an instability. *Journal of Theoretical Biology*, 26(3):399–415.

Kennedy, J. and Eberhart, R. C. (1995). Particle Swarm Optimisation. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, Piscataway, NJ, USA. IEEE service center.

Kennedy, J., Eberhart, R. C., and Shi, Y. (2001). *Swarm Intelligence*. Morgan Kaufmann, San Francisco, CA, USA.

Khajehpour, P., Lucas, C., and Araabi, B. N. (2005). Hierarchical image segmentation using ant colony and chemical computing approach. In Wang, L., Chen, K., and S. Ong, Y., editors, *Advances in Natural Computation*, volume 3611 of *Lecture Notes in Computer Science*, pages 1250–1258. Springer Berlin / Heidelberg.

Knox, J. (1994). Tabu search performance on the symmetric traveling salesman problem. *Computers & Operations Research*, 21(8):867–876.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480.

Kotsiantis, S. B. and Pintelas, P. E. (2004). Recent advances in clustering: A brief survey. *WSEAS Transactions on Information Science & Applications*, 1(1):73–81.

Krishnapuram, R. and Keller, J. M. (1993). A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98–110.

Labella, T. H., Dorigo, M., and Deneubourg, J.-L. (2006). Division of labor in a group of robots inspired by ants' foraging behavior. *ACM Transactions on Autonomous & Adaptive Systems*, 1(1):4–25.

Lai, J. Z. C. and Liaw, Y.-C. (2008). Improvement of the k-means clustering filtering algorithm. *Pattern Recognition*, 41(12):3677–3681.

Laine, A. F., Schuler, S., Fan, J., and Huda, W. (1994). Mammographic feature enhancement by multiscale analysis. *IEEE Transactions on Medical Imaging*, 13(4):725–740.

Lawler, E. L., Lenstra, J. K., Rinnooy-Kan, A. H. G., and Shmoys, D. B. (1985). *The traveling salesman problem*. John Wiley & Sons, Chichester, UK.

Lerman, K., Jones, C., Galstyan, A., and Matarić, M. J. (2006). Analysis of dynamic task allocation in multi-robot systems. *International Journal of Robotics Research*, 25(3):225–241.

Li, Q., Mitianoudis, N., and Stathaki, T. (2007). Spatial kernel k-harmonic means clustering for multi-spectral image segmentation. *IET Image Processing*, 1(2):156–167.

Liew, A. W. C., Leung, S. H., and Lau, W. H. (2000). Fuzzy image clustering incorporating spatial continuity. *IEE Proceedings - Vision, Image & Signal Processing*, 147(2):185–192.

Lu, D. S. and Chen, C. C. (2008). Edge detection improvement by ant colony optimization. *Pattern Recognition Letters*, 29(4):416–425.

Lumer, E. D. and Faieta, B. (1994). Diversity and adaptation in populations of clustering ants. In *Proceedings of the 3rd International Conference on Simulation of Adaptive Behavior: From Animals to Animats 3*, pages 501–508, Cambridge, MA, USA. MIT Press.

Ma, G. J., Duan, H. B., and Liu, S. Q. (2007). Improved ant colony algorithm for global optimal trajectory planning of UAV under complex environment. *International Journal of Computer Science & Applications*, 4(3):57–68.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In Le Cam, L. M. and Neyman, J., editors, *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability - Vol. 1*, pages 281–297. University of California Press, Berkeley, CA, USA.

Makowski, A. M. (2008). The binary bridge selection problem: Stochastic approximations and the convergence of a learning algorithm. In *Proceedings of the 6th international conference on Ant Colony Optimization and Swarm Intelligence*, ANTS '08, pages 167–178, Berlin, Heidelberg. Springer-Verlag.

Maniezzo, V. (1999). Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. *INFORMS Journal on Computing*, 11:358–369.

Marcano-Cedeño, A., Jevtić, A., Álvarez Vellisco, A., and Andina, D. (2009). New artificial metaplasticity mlp results on standard data base. In Cabestany, J., San-

doval, F., Prieto, A., and Corchado, J., editors, *Bio-Inspired Systems: Computational and Ambient Intelligence*, volume 5517 of *Lecture Notes in Computer Science*, pages 174–179. Springer Berlin / Heidelberg.

Martin, M., Chopard, B., and Albuquerque, P. (2002). Formation of an ant cemetery: Swarm intelligence or statistical accident? *Future Generation Computer Systems*, 18(7):951–959.

Matarić, M. J., Sukhatme, G. S., and Østergaard, E. H. (2003). Multi-robot task allocation in uncertain environments. *Autonomous Robots*, 14(2-3):255–263.

Melgar, I., Fombellida, J., Jevtić, A., and Seijas, J. (2009). Swarm architectures for ground-based air defense systems of systems. In *Proceedings of the 7th IEEE International Conference on Industrial Informatics, INDIN 2009*, pages 783–788.

Mertzios, B. G. and Tsirikolias, K. (2001). Applications of coordinate logic filters in image analysis and pattern recognition. In *Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis, 2001. ISPA 2001.*, pages 125–130.

Michael, N., Zavlanos, M. M., Kumar, V., and Pappas, G. J. (2008). Distributed multi-robot task assignment and formation control. In *Proceedings of the IEEE International Conference on Robotics and Automation, 2008. ICRA 2008.*, pages 128–133.

Miranda, V. and Fonseca, N. (2002). EPSO - evolutionary Particle Swarm Optimization, a new algorithm with applications in power systems. In *IEEE/PES Transmission and Distribution Conference and Exhibition 2002: Asia Pacific*, volume 2, pages 745–750.

Naka, S., Genji, T., Yura, T., and Fukuyama, Y. (2003). A hybrid Particle Swarm Optimization for distribution state estimation. *IEEE Transactions on Power Systems*, 18(1):60–68.

Nezamabadi-pour, H., Saryazdi, S., and Rashedi, E. (2006). Edge detection using ant algorithms. *Soft Computing*, 10(7):623–628.

Nguyen, H. D., Yoshihara, I., Yamamori, K., and Yasunaga, M. (2007). Implementation of an effective hybrid GA for large-scale traveling salesman problems. *IEEE Transactions on Systems, Man & Cybernetics, Part B: Cybernetics*, 37(1):92–99.

Ojeda-Magaña, B., Quintanilla-Domínguez, J., Ruelas, R., and Andina, D. (2009). Images sub-segmentation with the PFCM clustering algorithm. In *Proceedings of the 7th IEEE International Conference on Industrial Informatics, 2009. INDIN 2009.*, pages 499–503.

Pal, N. R., Pal, K., and Bezdek, J. C. (1997). A mixed c-means clustering model. In *Proceedings of the 6th IEEE International Conference on Fuzzy Systems*, pages 11–21.

Pal, N. R., Pal, K., Keller, J. M., and Bezdek, J. C. (2005). A possibilistic fuzzy c-means clustering algorithm. *IEEE Transactions on Fuzzy Systems*, 13(4):517–530.

Panetta, K. A., Wharton, E. J., and Agaian, S. S. (2008). Logarithmic edge detection with applications. *Journal of Computers*, 3(9):11–19.

Partridge, B. L. (1982). The structure and function of fish schools. *Scientific American*, 246(6):114–123.

Pham, D. T., Ghanbarzadeh, A., Koç, E., Otri, S., Rahim, S., and Zaidi, M. (2006). The bees algorithm – a novel tool for complex optimisation problems. In *Proceedings of the 2nd Virtual International Conference on Intelligent Production Machines and Systems, IPROMS 2006*, pages 454–461.

Pratt, W. K. (1991). *Digital Image Processing*. John Wiley & Sons, Inc., New York, NY, USA.

Prewitt, J. M. S. (1970). *Object Enhancement and Extraction in Picture Processing and Psychopictorics*. Academic Press, New York.

Quintanilla-Domínguez, J., Cortina-Januchs, M. G., Ojeda-Magaña, B., Jevtić, A., Vega-Corona, A., and Andina, D. (2010). Microcalcification detection applying artificial neural networks and mathematical morphology in digital mammograms. In *World Automation Congress, WAC2010*, pages 1–6.

Ramos, V. and Almeida, F. (2000). Artificial ant colonies in digital image habitats - a mass behavior effect study on pattern recognition. In *Proceedings of the ANTS'2000 - 2nd International Workshop on Ant Algorithms (From Ant Colonies to Artificial Ants), M. Dorigo, M. Middendorf, T. Stzle (Eds.)*, pages 113–116, Brussels, Belgium.

Ramos, V., Muge, F., and Pina, P. (2002). Self-organized data and image retrieval as a consequence of inter-dynamic synergistic relationships in artificial ant colonies. In Ruiz-del Solar, J., Abraham, A., and M., K., editors, *Frontiers in Artificial Intelligence and Applications, Soft Computing Systems - Design, Management and Applications*, volume 87 of *2nd International Conference on Hybrid Intelligent Systems*, pages 500–509. Springer Berlin / Heidelberg, Santiago, Chile.

Rana, O. F. and Stout, K. (2000). What is scalability in multi-agent systems? In *Proceedings of the 4th International Conference on Autonomous Agents*, AGENTS '00, pages 56–63, New York, NY, USA. ACM.

Ray, A. K., Benavidez, P., Behera, L., and Jamshidi, M. (2009). Decentralized motion coordination for a formation of rovers. *IEEE Systems Journal*, 3(3):369–381.

Reinelt, G. (1991). TSPLIB–a traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384.

Resnick, M. (1994). *Turtles, termites, and traffic jams: Explorations in massively parallel microworlds*. MIT Press, Cambridge, MA, USA.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Computer Graphics*, 21:25–34.

Richardson, A. J., Risien, C., and Shillington, F. A. (2003). Using self-organizing maps to identify patterns in satellite imagery. *Progress in Oceanography*, 59:223–239.

Rosenkrantz, D. J., Stearns, R. E., and Lewis, P. M. (2009). An analysis of several heuristics for the traveling salesman problem. In Ravi, S. S. and Shukla, S. K., editors, *Fundamental Problems in Computing*, pages 45–69. Springer Netherlands.

Runkler, T. A. (2005). Ant colony optimization of clustering models: Research articles. *International Journal of Intelligent Systems*, 20:1233–1251.

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A modern approach*. Prentice Hall, New Jersey, USA, 2nd edition.

Saatchi, S. and Hung, C. C. (2005). Hybridization of the ant colony optimization with the k-means algorithm for clustering. In Kalviainen, H., Parkkinen, J., and Kaarna, A., editors, *Image Analysis*, volume 3540 of *Lecture Notes in Computer Science*, pages 511–520. Springer Berlin / Heidelberg.

Schwager, M., McLurkin, J., Slotine, J.-J., and Rus, D. (2009). From theory to practice: Distributed coverage control experiments with groups of robots. In Khatib, O., Kumar, V., and Pappas, G., editors, *Experimental Robotics*, volume 54 of *Springer Tracts in Advanced Robotics*, pages 127–136. Springer Berlin / Heidelberg.

Shell, D. A. and Matarić, M. J. (2003). On the use of term "stigmergy". In *Poster Paper in Proceedings of the 2nd International Workshop on the Mathematics & Algorithms of Social Insects*, page 193.

Shi, Y. and Eberhart, R. C. (2001). Fuzzy adaptive Particle Swarm Optimization. In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 1, pages 101–106.

Shih, F. Y. and Cheng, S. (2004). Adaptive mathematical morphology for edge linking. *Information Sciences–Informatics & Computer Science: An International Journal*, 167(1-4):9–21.

Shin, M. C., Goldgof, D. B., and Bowyer, K. W. (2001). Comparison of edge detector performance through use in an object recognition task. *Computer Vision & Image Understanding*, 84(1):160–178.

Sobel, I. and Feldman, G. (1968). A 3×3 isotropic gradient operator for image processing. *presented at a talk at the Stanford Artificial Project, unpublished but often cited.*

Stützle, T. and Hoos, H. H. (2000). $\mathcal{MAX} - \mathcal{MIN}$ Ant System. *Future Generation Computer Systems*, 16(9):889–914.

Tan, S. C., Ting, K. M., and Teng, S. W. (2006). Reproducing the results of ant-based clustering without using ants. In *IEEE Congress on Evolutionary Computation, 2006. CEC 2006.*, pages 1760–1767.

Teodorović, D. and Dell'Orco, M. (2005). Bee colony optimization - a cooperative learning approach to complex transportation problems. In *Advanced OR & AI Methods in Transportation*, Proceedings of 10th Meeting of the EURO Working Group on Transportation, page 5160.

Tian, J., Yu, W., and Xie, S. (2008). An Ant Colony Optimization algorithm for image edge detection. In *Proceedings of the IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence)*, pages 751–756, Hong Kong, China.

Tibshirani, R., Walther, G., and Hastie, T. (2000). Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistical Society, Series B*, 63:411–423.

Tsai, C.-F., Wu, H.-C., and Tsai, C.-W. (2002). A new data clustering approach for data mining in large databases. In *Proceedings of the International Symposium on Parallel Architectures, Algorithms & Networks, 2002. I-SPAN '02.*, pages 278–283.

Turing, A. M. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 237(641):37–72.

Vizine, A. L., De Castro, R. N., Hruschka, E. R., and Gudwin, R. R. (2005). Towards improving clustering ants: An adaptive ant clustering algorithm. *Informatica*, 29:143–154.

Von Frisch, K. (1967). *The Dance Language and Orientation of Bees.* Harvard University Press, Cambridge, MA.

Wang, J.-H. and Rau, J.-D. (2001). Vq-agglomeration: A novel approach to clustering. *IEE Proceedings - Vision, Image & Signal Processing*, 148(1):36–44.

Wedde, H. F., Farooq, M., and Zhang, Y. (2004). Beehive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. In *Ant Colony, Optimization and Swarm Intelligence*, volume 3172 of *Lecture Notes in Computer Science*, pages 83–94. Springer-Verlag.

Wei, L., Sheng, L., Yi, R. X., and Peng, D. (2008). A new contour detection in mammogram using sequential edge linking. In *Proceedings of the 2008 Second International Symposium on Intelligent Information Technology Application (IITA '08)*, volume 1, pages 197–200, Hong Kong, China.

Wenner, A. M. and Wells, P. H. (1990). *Anatomy of a controversy: The question of a "language" among bees.* Columbia University Press, New York.

Whitley, L. D., Starkweather, T., and Fuquay, D. (1989). Scheduling problems and traveling salesmen: The genetic edge recombination operator. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, pages 133–140, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Yang, X.-S. (2005). Engineering optimizations via nature-inspired virtual bee algorithms. In *Artificial Intelligence and Knowledge Engineering Applications: A*

Bioinspired Approach, volume 3562 of *Lecture Notes in Computer Science*, pages 317–323.

Yu, Z., Yu, W., Zou, R., and Yu, S. (2009). On ACO-based fuzzy clustering for image segmentation. In *Proceedings of the 6th International Symposium on Neural Networks: Advances in Neural Networks - Part II*, ISNN 2009, pages 717–726, Berlin, Heidelberg. Springer-Verlag.

Zhang, W.-J. and Xie, X.-F. (2003). DEPSO: Hybrid particle swarm with differential evolution operator. In *Proceedings of the IEEE International Conference on Systems, Man & Cybernetics*, volume 4, pages 3816–3821.

Zhenhua, W., Weiguo, Z., Jingping, S., and Ying, H. (2008). UAV route planning using multiobjective Ant Colony System. In *2008 IEEE Conference on Cybernetics and Intelligent Systems*, pages 797–800.